2017

COMPARATIVE STUDIES OF DIFFUSION MODELS AND ARTIFICIAL NEURAL INTELLIGENCE ON ELECTROCHEMICAL PROCESS OF U AND Zr DISSOLUTIONS IN LiCl-KCl EUTECTIC SALTS

Samaneh Rakhshan Pouri

COPYRIGHT PAGE

COMPARATIVE STUDIES OF DIFFUSION MODELS AND ARTIFICIAL NEURAL INTELLIGENCE ON ELECTROCHEMICAL PROCESS OF U AND Zr DISSOLUTIONS IN LiCl-KCl EUTECTIC SALTS

A Dissertation Submitted in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy at Virgina Commonwealth University

By

Samaneh Rakhshan Pouri,

Mechanical and Nuclear Engineering

Major Professor: Supathorn Phongikaroon, Ph.D.

Virginia Commonwealth University

Richmond, Virginia

July, 2017

Acknowledgments

This dissertation would never have been possible without the guidance of my advisor, and support from my husband, family, relatives, and friends. First, I wish to sincerely express my gratefulness and thankfulness to my lovely husband, M.D. Majid Jahromi, for his love, solid support, encouragement, and insightful comments to finish this study. Most importantly, I would like to send my gratefulness and love to my parents, Fataneh and Davoud Rakhshan Pouri, whose experiences all of the ups and downs of my research. They are my ultimate role models whose love, guidance and encouragement are with me in whatever I pursue. I can't be more proud to be their daughter. Words cannot express the gratitude that my husband and parents deserve for all their contributions.

I would like to express my sincere gratefulness and appreciation to my advisor, Dr. Supathorn (Supy) Phongikaroon, for his support, guidance, patience, and understanding. He spent a lot of time for training and mentoring me not only in pursuing the Ph.D. but also in surviving in a new country. He also provided me an excellent atmosphere for doing my research, and supporting my attendance at various nuclear engineering conferences.

Additionally, I would like to acknowledge my committee members, Drs. Sama Bilbao Y Léon, Karla Mossi, Carl R Elk, and Prof. Milos Manic for their invaluable inputs, and interest in my work. I also would like to thank Dalsung Yoon, Ammon N Williams, Riyadh Motny, Dumidu Shanika Wijayasekara and my other colleagues for their help, and commonts.

I would like to thank my aunt and uncle, Mrs. & Dr. Mehraban, for all their moral support, suggestions and encouragement. Additionally, I owe to my younger brother, Saeed, for not being

present and accompanying in his new life chapter. He always is cheering me up and teaching me to take the life easy.

I would like to devote this dissertation to my grandmother who is not living in this world anymore. It was one of her dreams to participate in my graduation ceremony. I loved her so much and I am sure she is so proud now.

I am immensely grateful to you all and words cannot express my gratefulness and appreciation.

Contents

List of Figures

List of Tables

List of Abbreviations and Symbols

Abbreviations and Acronyms

Name	Description
ANI	Artificial Neural Intelligent
ANL	Argonne National Laboratory
ASV	Anodic Stripping Voltammetry
BRA	Bayesian Regularization algorithem
CP	Chronopotentiometry
CV	Cyclic Voltammetry
EBR-II	Experimental Breeder Reactor-II
ER	Electrorefiner
FDB	Fuel Dissolution Basket
FP	Fission Product
GUI	Graphic User Interface
HLW	High Level Waste
IFR	Integral Fast Reactor
INL	Idaho National Laboratory
LCC	Liquid Cadmium Cathode
PUREX	Plutonium Uranium Redox Extraction
MA	Minor Actinide

TRU	Transuranic
UNF	Used Nuclear Fuel

Symbols

Name	Description	Unit
A	Surface area of working electrode	cm^2
C_o^*	Bulk concentration of oxidized species	mol/cm^3
$C_{oAirrev}$	Concentration of oxidant species at anodic irreversible side	mol/cm^3
C_{oArev}	Concentration of oxidant species at anodic reversible side	mol/cm^3
$C_{oCirrev}$	Concentration of oxidant species at cathodic irreversible side	mol/cm^3
C_{oCrev}	Concentration of oxidant species at cathodic reversible side	mol/cm^3
C_R^*	Bulk concentration of reductant species	mol/cm^3
$C_{RAirrev}$	Concentration of reductant species at anodic irreversible side	mol/cm^3
C_{RArev}	Concentration of reductant species at anodic reversible side	mol/cm^3
$C_{RCirrev}$	Concentration of reductant species at cathodic irreversible side	mol/cm^3
C_{RCrev}	Concentration of reductant species at cathodic reversible side	mol/cm^3
D_R	Diffusion coefficient of reduced species	cm^2/s
D_o	Diffusion coefficient of oxidized species	cm^2/s
E^{o*}	Apparent standard potential	V
E_i	Initial potential	V

E°	Formal electrode potential	V
F	Faraday's constant	C/eq
i	Current	amp/cm^2
j	Serial number of the subinterval in integral steps	
k_s	Standard rate constant	cm/s
n	Number of electron transferred per mole	eq/mol
R	Universal gas constant	J/mol·K
t	Time	s
T	Absolute temperature	K
x	Linear distance from the electrode surface	cm

Greek

Name	Description	Unit
α	Transfer coefficient	
σ	Length of subinterval	
ε	The time that the scan is reversed	s
ν	Scan rate	V/s

Abstract

COMPARATIVE STUDIES OF DIFFUSION MODELS AND ARTIFICIAL NEURAL INTELLIGENCE ON ELECTROCHEMICAL PROCESS OF U AND Zr DISSOLUTIONS IN LiCl-KCl EUTECTIC SALTS

By Samaneh Rakhshan Pouri, Ph.D.

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy at Virginia Commonwealth University.

Virginia Commonwealth University, 2017.

Major Professor: Supathorn Phongikaroon, Associate Professor of Mechanical and Nuclear Engineering Department.

The electrorefiner (ER) is the heart of pyroprocessing technology operating at a high-temperature (723 K – 773 K) to separate uranium from Experimental Breeder Reactor-II (EBR-II) used metallic fuel. One of the most common electroanalytical methods for determining the thermodynamic and electrochemical behavior of elemental species in the eutectic molten salt LiCl-KCl inside ER is cyclic voltammetry (CV). Information from CV can possibly be used to estimate diffusion coefficients, apparent standard potentials, transfer coefficients, and numbers of electron transferred. Therefore, predicting the trace of each species from the CV method in an absence of experimental data is important for safeguarding this technology. This work focused on the development an interactive computational design for the CV method by analyzing available uranium chloride data sets (1 to 10 wt%) in a LiCl-KCl molten salt at 773 K under different scan rates to help elucidating, improving, and providing robustness in detection analysis. A principle method and a computational code have been developed by using electrochemical fundamentals

and coupling various variables such as: the diffusion coefficients, formal potentials, and process time duration. Although this developed computational model works moderately well with reported uranium data sets, it experiences difficulty in tracing zirconium data sets due to their complex CV structures. Therefore, an artificial neural intelligent (ANI) data analysis has been proposed to resolve this issue and to provide comparative study to the precursor computational modeling development. For this purpose, ANI has been applied on 0.5 to 5 wt% of zirconium chloride in LiCl-KCl eutectic molten salt at 773 K under different scan rates to mimic the system and provide current and potential simulated data sets for the unseen data. In addition, a Graphical User Interface (GUI) through the commercial software *Matlab* was created to provide a controllable environment for different users. The computational code shows a limitation in high concentration CV prediction, capturing the adsorption peaks, and provides a dissimilarity. However, the model is able to capture the important anodic and cathodic peaks of uranium chloride CV which is the main focus of this study. Furthermore, the developed code is able to calculate the concentration of each species as a function of time. Due to the complexity of the CV of zirconium chloride, the computational model is used to predict the probability reactions occurring at each peak. The resulting study reveals that the reaction at the highest anodic peak is related to the combination of 70% Zr/Zr^{+4} and 30% Zr/Zr^{+2} for the 1.07 wt% and 2.49 wt% zirconium chloride and 30% Zr/Zr^{+4} and 70% Zr/Zr^{+2} combination for 4.98 wt% $ZrCl_4$. The proposed alternative ANI method has demonstrated its capability in predicting the trend of species in a new situation with a high accuracy on predictions without any dissimilarity. Two final structures from zirconium chloride study which high accuracy (that is, a low error) are related to [9, 15, 10]-18 and [10, 11, 25]-19. These two final structures have been applied on uranium chloride salt experimental data sets to further validate the ANI's ability and concept. Three different fixed data combinations were considered. The result indicates

that by increasing the number of training data sets it does not necessarily help improving the prediction process. ANI implementation outcome on uranium chloride data set illustrates a good prediction with a specific fixed data combination and [9, 15, 10]-18 structure. Thus, it can be concluded that ANI is a promising method for safeguarding pyroprocessing technology due to its robustness in predicting the CV plots with high accuracy.

Chapter 1: Introduction

1.1 Purpose

Pyroprocessing (also known as ‘pyrochemical, electrometatullurgical, and/or electrochemical’) technology, which was developed by Argonne National Laboratory (ANL), is a high-temperature reprocessing method of Experimental Breeder Reactor-II (EBR-II) used nuclear fuel (UNF). The heart of this process is an electrorefiner (ER) which contains different fission, rare-earth, and transuranic chloride compositions during the operation. This is still a developing technology that needs to be advanced for the commercial reprocessing design of UNF. As a result, it is important to gain information knowledge within the ER in terms of intelligent materials detection and accountability towards safeguards to boost this technology. The main goal of this study is to develop a near real time monitoring detection program to trace the trend of each species and predict the unseen situation toward pyroprocessing safeguards. For this purpose, a diffusion model is first developed in *Matlab* software to predict the cyclic voltammetry (CV) of uranium chloride in different concentrations and scan rates in a very short time. To provide a compatible model with other transuranic material such as zirconium chloride, a novel electrochemical data analysis and simulation using an artificial neural intelligent (ANI) method has been proposed and developed as the next step. Such intelligent signal detection requires understanding of massive ER systemic parameters through available electrochemical data sets. The advantage of ANI approach is that it could be trained to mimic the system by driving the data sets interrelation between variables to provide current and potential simulated data sets for the unseen situation with a high accuracy of prediction.

The diffusion and ANI modeling methods drive the ultimate goal of this work, which is to provide a comparative study between both techniques to illustrate and deliver the best methodology for robust concentration detection and measurement from CV graphs. The selected method will be an important tool that is applicable for safeguarding applications in pyroprocessing technology.

1.2 Motivation

The composition analysis in ER can be measured by experimental techniques such as inductively coupled plasma mass spectrometry (ICP-MS) which is a common practice, inductively coupled plasma atomic emission spectrometry (ICP-AES), and Raman Spectroscopy. However, the extraction process, material transfer, and sample preparation may take up to approximately 3 weeks due to radiation transferred process from the main operating facility to radiochemistry laboratory and analytical preparation routines which doesn't fulfill the near real time monitoring goal (Ref. 1). Therefore, laser-induced breakdown spectroscopy (LIBS), Ultraviolet-visible spectroscopy (UV-Vis), and electrochemistry techniques (Cyclic Voltammetry, Chronopotentiometry, anodic stripping voltammetry) have been proposed as an alternative techniques (See Fig. 1.1) through the funding supports from the Department of Energy – Nuclear Energy University Program. These techniques can monitor the behavior of ER contains in the microsecond to 10 minutes; however, they still have some difficulties and are under developments. One of the experimental electrochemical method which has been broadly utilized to measure thermodynamic behavior of uraniium and zirconium within ER is CV (Refs. 2 to 4). There are diverse CV software packages, which can provide the current versus potential diagram such as

Bio-logic EC-Lab and Power CV (Ref.5). However, predicting the trace of species without experimental data sets in a relatively short time has become a huge concern and a great need in nuclear material detection and accountancy (Refs. 5 and 6). Although some software packages (for example, BASi DigiSim Simulation) have been developed to identify the species, analyses using them may require many hours to obtain the final outputs defeating the original purpose of a real time detection intention. In addition, there is an analytical cyclic voltammetry study which uses a nonlinear least square procedure to fit a BET (multilayer adsorption) model on the experimental data to trace the species. However, this study is limited to a high standard reaction rate and similarity oxidant and reductant diffusivity. Also, the diffusivities calculated with the BET model are considerably larger than those with the Delahay equation (Refs. 7 and 8). The aforementioned issues provide a motivation for this study.

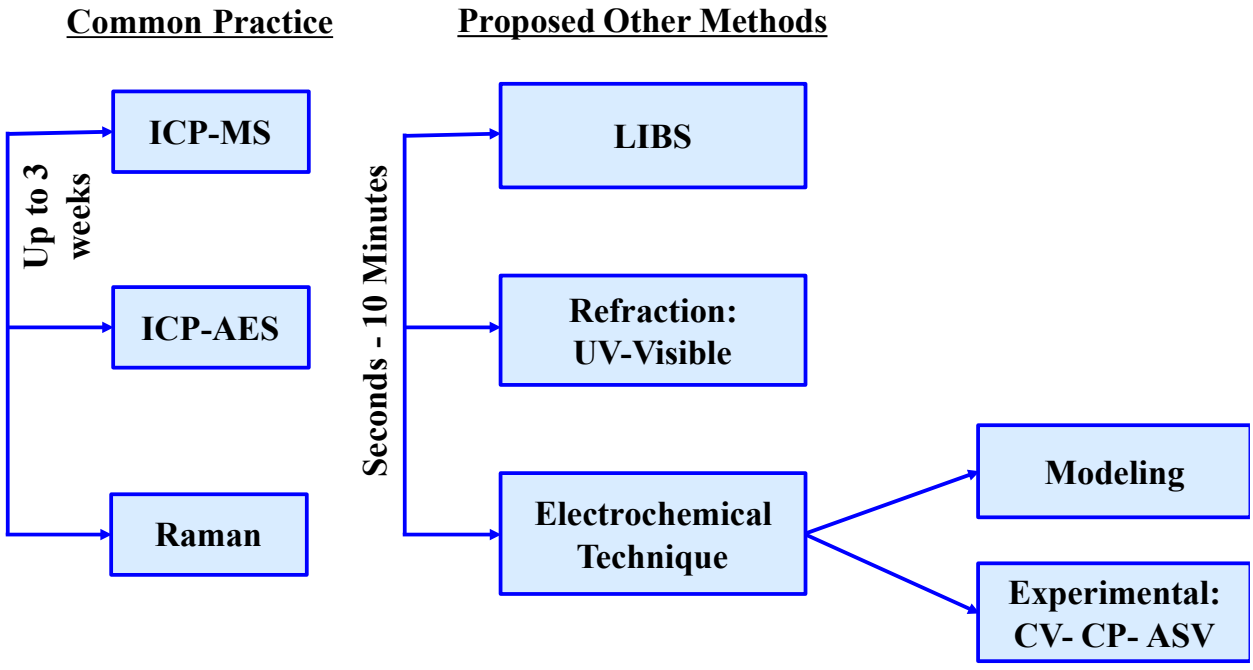

Fig. 1.1 Analytical method for material analysis.

1.3 Approach

Three phases are established in this research study. In Phase I, extensive literature review has been conducted to find the uranium properties such as diffusion coefficients, and apparent standard potentials in the electrorefining system reported by other researchers. In addition, a depth study on numerical modeling of CV for zirconium chloride ($ZrCl_4$) and uranium chloride (UCl_3) is accomplished in this phase. Therefore, a modeling design of uranium chloride with implementing the experimental data sets has been developed. The methodology is accomplished by reversely solving the essential equations via Laplace transform through the incorporation of electrochemical values existing in published literature at a very low standard rate constant, and reported data by Hoover (Ref. 6) on the electrochemical study of uranium chloride (1 to 10 wt%) in LiCl-KCl eutectic salt at 773 K. All experimental data sets from Ref. 6 have been utilized in this study to find the diffusion coefficients, and formal potentials based on the initial inputs reported at literatures. Hence, the predicted current versus potential graph and concentration at different time graph can be compared to results reported in the literature (Ref. 6 and 9). Furthermore, a Graphical User Interface (GUI) has been implemented allowing an individual user to control directly and make adjustments to support material signal detection and analysis (Ref. 10). Several cases were investigated demonstrating the range of appropriateness and determining the acute conditions at which a better compatibility of modified diffusion model is effective. In addition, to extent the model's potential, Zr, which is one of the major components in EBR-II used metallic fuel, and its CVs (0.5 to 5 wt% of zirconium chloride in LiCl-KCl eutectic molten salt at 773 K under different scan rates (Ref. 6) have been tested. Due to the complexity in Zr cyclic

voltammogram data sets, the modified model reaches its limits in providing the accurate predictions.

Therefore, in Phase II, ANI has been proposed as another novel data analysis tool providing a simulation method that can be applied on massive experimental data sets. The main goal of this phase is to train the computer by feeding the massive data sets collected by Hoover (Ref. 6), over 230,000, to predict the cyclic voltammetry of zirconium chloride (0.5 to 5 wt%). For this purpose, ANI has been implemented on the massive data sets through iteration and interrelationships among system variables such as scan rate, current, potential, process time, and weight percent. The approach of this phase is determining the number of hidden layer (1 to 3), neurons at each layer (1 to 30), validation checks (1 to 30) and the minimum number of training data set requirement for achieving the lowest error. To simplify the comparison of the results, the average percent error between experimental and predicted data sets for 0.5 wt% at 200 and 450 mV/s has been considered. At the end, a GUI has been prepered to make the ANI environment easier for the user. The user can provide input such as the desire number of layer, neurons, and validation checks to get the error table and cyclic voltammetry plot for the desire concentration at different scan rates.

In Phase III, ANI is applied on massive experimental data sets of 0.5 to 10 wt% of uranium chloride (around 354,000 points) in LiCl-KCl eutectic molten salt at 773 K collected by Hoover (Refs. 6, and 9). The same approach is repeated at this stage to find a structure that provides a low error for uranium chloride. The number of hidden layers, neurons, and validation checks which were chosen for zirconium chloride are tested for uranium chloride. The predicted CV by ANI are compared with experimental data set and is cross-validated with the diffusion model results in Phase I to find a best method of signal detection.

These phases were planned, explored and accomplished at Mechanical and Nuclear Engineeering Department of Virginia Commonwealth University (VCU) for the 2.5-year period. Table 1.1 describes the timeframe to accomplish this study.

Table 1.1 Schedule and time frame for accomplishment of this project.

	Year 1 (2015)				Year 2 (2016)				Year 3 (2017)			
Phase	1	2	3	4	1	2	3	4	1	2	3	4
I												
II												
III												

1.4 Organization of the Dissertation

The first step of this dissertation is to develop a robust computational model for fitting the uranium chloride CV. The developed diffusion model experienced some limitations for more complicated CV such as zirconium choride. Therefore, ANI was implemented on massive experimental data set of zirconium chloride to predict the current versus voltage plots. Furthermore, the ANI was applied on the uranium data sets to verify the ANI's concept. At the end, two methods were compared together and the results illustrated that ANI was a best methodology for a near real time concentration detection and measurement for CV graphs.

This dissertation is consisted of six chapters. In Chapter 1, purpose, motivation, and approach of this study has been discussed. Chapter 2 contains the literature survey while focusing on pyroprocessing technology, electrorefiner, and cyclic voltammetry. In Chapter 3, the diffusion model has been explained and calculation of the diffusion coefficient and the apparent standard potentials for zirconium and uranium chloride based on the Hoover experimental data sets are

being delivered. In addition, numerical method, fundamental equations, and computational process of diffusion model has been explained. Furthermore, the CV plots of diffusion model for both uranium and zirconium chloride are compared with collected experimental data sets. Theories and computational procedures of ANI are given in Chapter 4. In addition, ANI results related to the error comparison of first to third hidden layers, and CV comparison of ANI predicted with actual experimental data for zirconium are illustrated in this chapter. The final structures have been implemented on the uranium experimental data sets to determine if ANI method is capable of applying on different material data sets. Chapter 5 provides the summary of the diffusion model and ANI being compared together and offers a best method for safeguarding pyroprocessing technology. In this final chapter, the dissertation key points are summarized with suggestions for the future work as well.

Chapter 2: Review of Electrochemical Process

2.1 Pyroprocessing

It can be stated that one of the significant source of electricity production in worldwide is nuclear power because there are 441 Commercial nuclear power plant in 2016 with a global generation capacity of 382.9 GW(e) (Ref. 11). The nuclear power is expanding due to increasing the number of energy demand. One of the main controversial issue related to the nuclear power production in terms of political, economic, and social concern is related to the used nuclear fuel (UNF) management (Ref. 12). It is very important to recover components from the used fuel to save the fuel resources and to solve the storage capacity issue. In fact, there are 266,000 tHM stored used fuel in 2016 with accumulating rate of 7000 tHM/year reprocessing capacity (Ref. 11). Therefore, reprocessing of UNF while contains 96% of uranium is very crucial to minimize the volume of radioactive waste and minimize the need for uranium sources (Ref. 13).

Two methods that have been implemented for reprocessing technology are referred as aqueous and dry separations. Plutonium Uranium Redox Extraction (PUREX) is one of the well-known aqueous process to recover plutonium and uranium from UNF through chemical separations and several cycles of solvent extractions by implementing highly contrated nitric acid (Refs. 12, 13, and 14). Fig. 2.1 illustrates shematic of PUREX flowsheet (Ref. 15). Since PUREX has a potential to get utilized in weapon-grade materials, pyroprocessing, which is another well-developed dry process technique, has been proposed and considered as an alternative option in material recovery (especially, to overcome the proliferation concern) (Refs. 12, and 13, and 16).

In addition, pyroprocessing can be designed in compact facilities and decrease the risk of UNF transportation (Ref. 14).

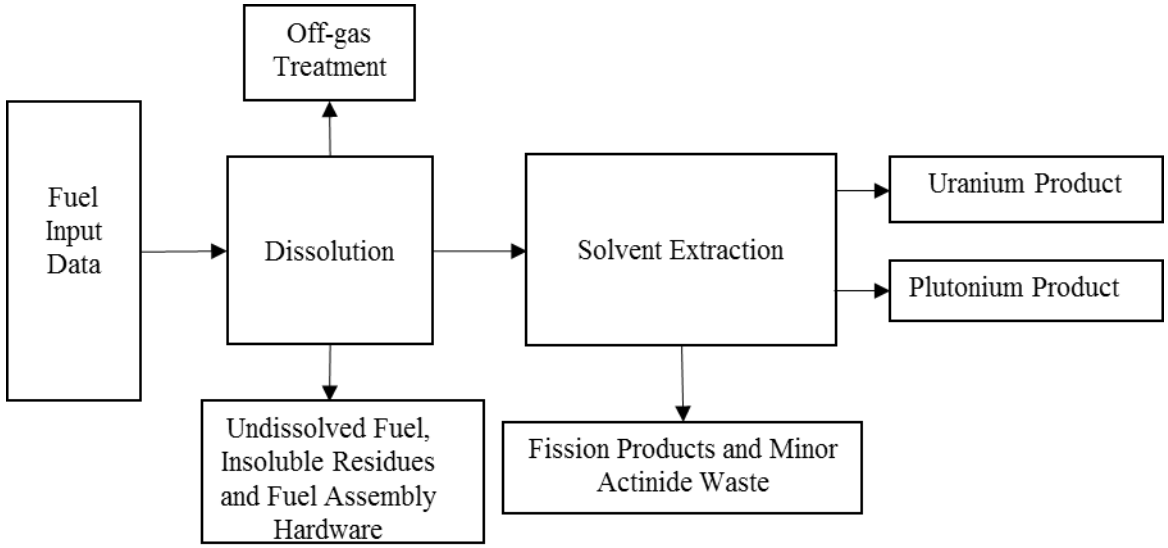

Fig. 2.1 Schematic of PUREX flowsheet (Ref. 14).

Pyroprocessing technology, known as electrochemical process, electrometallurgical reprocessing, or pyrochemial technology, was originally developed during the Integral Fast Reactor (IFR) program by Argonne National Laboratory (ANL) about 1984 to 1995 (Refs. 16, 17, and 18). This technology is a high-temperature (T > 723 K) reprocessing of used metallic nuclear fuel from Experimental Breeder Reactor- II (EBR-II). Currently, pyroprocessing has been operated at the Idaho National Laboratory (INL) with two missions; (1) treatment of EBR-II spent fuel, and (2) improvement of advanced technology for nuclear fuel cycle and commercialization (Ref. 19). The media at pyroprocessing is molten salt electrolyte rather than aqueous solusions and solvents (Ref. 18). Fig. 2.2 displays the pyroprocess shematic (Ref. 16).

As Fig. 2.2 illustrates, the electrorefiner (ER) is the heart of pyroprocessing. Uranium (U) and other transuranic (TRU) elements are collected for rods fabrication as a new fuel (Ref. 19). In

addition, the high level waste (HLW) components are converted into ceramic and metallic waste forms (Ref. 19). The elements from the ER go in a high temperature vacuum furnace (1473 K, 27 Pa) in cathode processor to remove and evaporate adhered salt or cadmium and to produce the pure metal (Refs. 12, and 19). The ingot product will then be injected in a casting furnace to fabricate new metal fuels (Ref. 12). Furthermore, the electrolyte salt with TRU, fission products (FPs) and NaCl from the ER are immobilized into ceramic waste forms by removing actinides and FPs through V-Blender (Ref. 12). Moreover, residual cladding hull, zirconium (Zr) and noble metals in the anodic basket are processed into metal waste furnace to be disposed as metal waste form (Refs. 12, and 19).

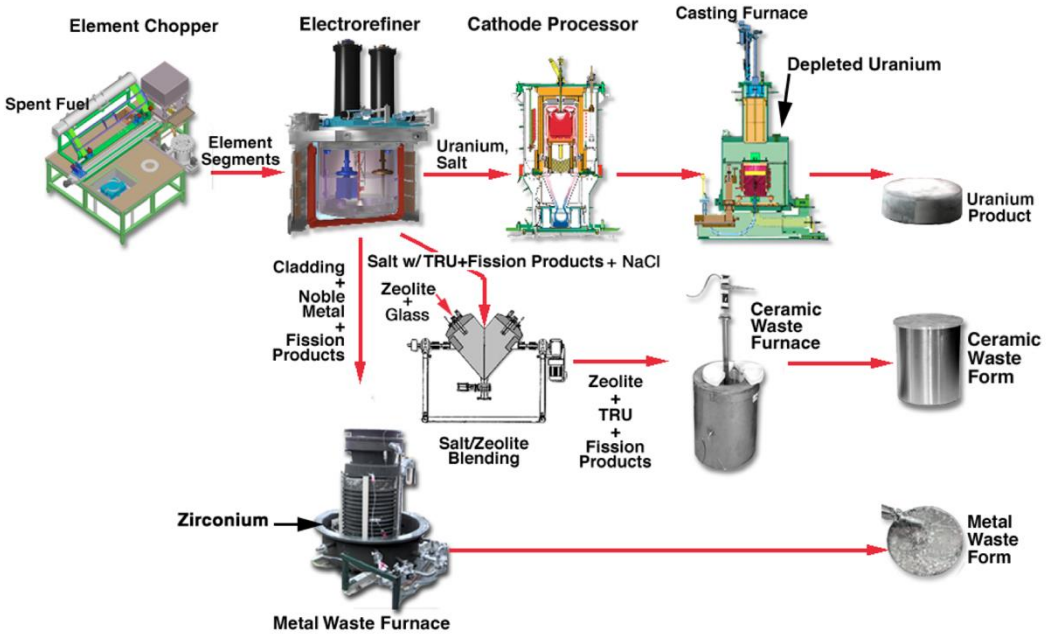

Fig. 2.2 Pyroprocessing technology for the used nuclear fuel treatment (Ref. 16).

2.2 Electrorefiner (ER)

The main key unit operation in pyroprocesssing is an electrorefiner (ER) with dynamic compositions of molten salt during the process (Ref. 16). There are two engineering ERs at INL,

called Mark-IV and Mark-V, to recover uranium from cladding hulls, bond sodium, and noble metals in the spent fuel of EBR-II (Ref. 19). Fig. 2.3 illustrates drawing of these systems which have the similar size but different markedly anode and cathode electrode designs (Ref. 12, and 19). The Mark-IV ER has been used to treat the highly enriched uranium (driver fuels with about 63% of U-235) and Mark-V ER operates with the depleted uranium (blanket fuel). In addition, the Mark-IV has a molten cadmium cathode whereas the Mark-V does not have any liquid cathode. Manifold number of studies have been performed on Mark-IV ER and almost 830 kgHM of driver fuel has been processed in this electrorefiner (Refs. 12, and 19).

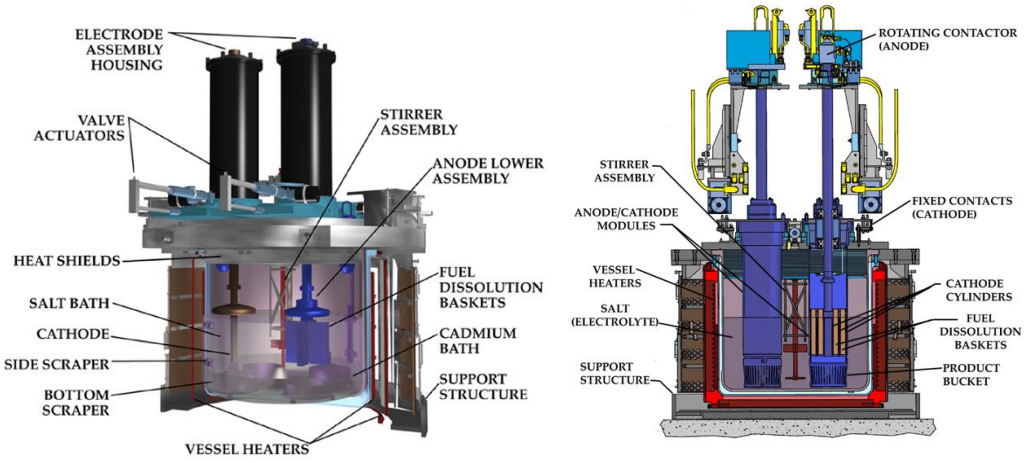

Fig. 2.3 Mark-IV (Left) and Mark-V (Right) electrorefiners (Ref. 19).

The electrolyte of ER consisted of a molten salt, typically LiCl-KCl (44.2 wt% LiCl, 55.8 wt% KCl) (Ref. 18). The chopped highly enriched, metallic uranium-zirconium alloy in stainless steel cladding driver fuels are loaded into four stainless steel rectangular containers, called anodic fuel dissolution baskets (FDBs, see Fig. 2.4) (Refs. 20, and 21). Each anode assembly can hold 8 kg of uranium and rotates to provide a homogenous mass transport (Ref. 22).

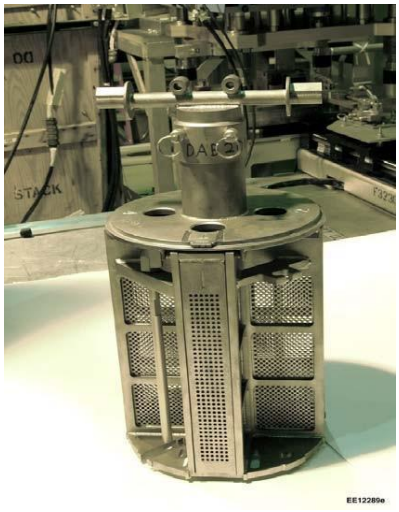

Fig. 2.4 Anodic dissolution basket (Ref. 21).

After basket loading into the molten salt, the reduction between bond sodium and active metal occures (Refs. 21, and 22). U, Pu, minor actinides (MAs), and rare earth materials are dissolved in the molten LiCl-KCl salt at 773 K. Only pure uranium is recoverd at stainless steel cathodic side by controlling applied voltage which called direct transport (Refs. 12, and 20). Fig. 2.5 indicates an example of deposited uranium. The residual U, Pu, and MAs are simultaneously collected by a liquid cadmium cathode (LCC) and called deposition (Ref. 12).

Fig. 2.5 Deposited uranium on the cathode electrode (Ref. 18).

Generally, the main reactions in an electrorefiner can be described as (Ref. 12):

Anode: $$U\ (anode) \rightarrow U^{+3}(salt) + 3e^{-} \quad (2.1)$$

Cathode: $$U^{3+}(salt) + 3e^{-} \rightarrow U(cathode) \quad (2.2)$$

Net Reaction: $$U(anode) \rightarrow U(cathode) \quad (2.3)$$

2.3 Cyclic Voltammetry (CV)

One of the popular methods for electrochemical reaction analysis is a cyclic voltammetry (CV) due to its straightforwardness as a part of measuring apparent standard potentials, transfer coefficients, numbers of electrons transferred, and diffusion coefficients (Refs. 1 to 3). In spite of the fact that there are different strategies for measuring thermodynamic behaviors such as chronopotentiometry (CP), anodic stripping voltammetry (ASV), etc., CV is the one that has overcome several disadvantages from other methods due to its simplicity in setup and fast response time aside from its function to directly evaluate reversibility and irreversibility for both the anodic and cathodic reactions (Refs. 4, and 23). The diffusion coefficient, apparent standard potential, transfer coefficient, equilibrium potential, and other parameters can also be determined through this method via different mathematical manipulations (Refs. 1 to 3).

The potentiostat to perform CV is consisted of three electrodes; working, reference, and counter (Ref. 24). The applied voltage on the working electrode has been linerely scanned to the negative direction and the potential direction reversed to the positive direction at the specific time with a constant scan rate, ν (Refs. 3, 12, and 25). In addition, the current between working and counter electrode are recorded. The reduction and oxidation reactions due to the potential changes

cause depletion occurrence at the electrode surface resulting in anodic and cathodic peaks (Ref. 9, and 12). There is a recording device to record the CV result as current versus potential graph which is shown in Fig. 2.6 (Ref. 25).

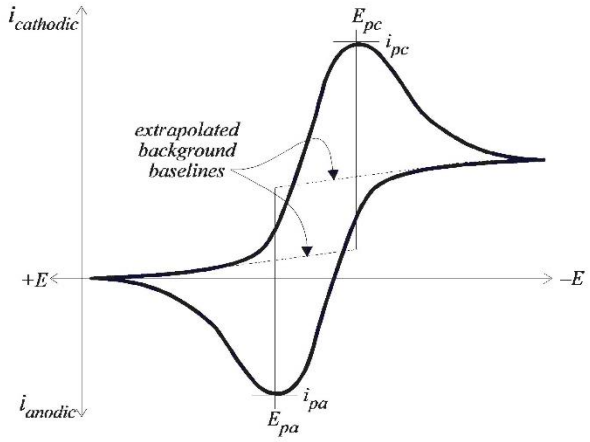

Fig. 2.6 Typical cyclic voltammogram plot (Ref. 25).

The formal potential (E°) can be calculated by averaging the cathodic and anodic potential peaks (E_{pc} and E_{pa}, respectively),

$$E^{\circ} = \frac{E_{pc} + E_{pa}}{2} \tag{2.4}$$

Cyclic voltammetry for 1wt% of UCl_3 at different scan rates for pure LiCl-KCl eutectic at 773K is illustrated in Fig. 2.7 (Ref. 9). Each peak is related to a specific oxidation and reduction reaction. Fig. 2.7 shows that while the potential is scanned from 0.0 V in the negative direction, the first reaction in peak A_c at -0.5 V is related to U^{+4} reduction. By further scanning, U^{+3} is absorbed to the working electrode surface in peak B_c at -1.5 V and then U^{+3} reduced to uranium metal in peak C_c. At approximately -2.4 V, the potential scan is reversed and uranium is oxidized to U^{+3} at the

first oxidation peak (C_a). Peak B_a represents the adsorption and A_a corresponds the U^{+3} reduction to U^{+4}.

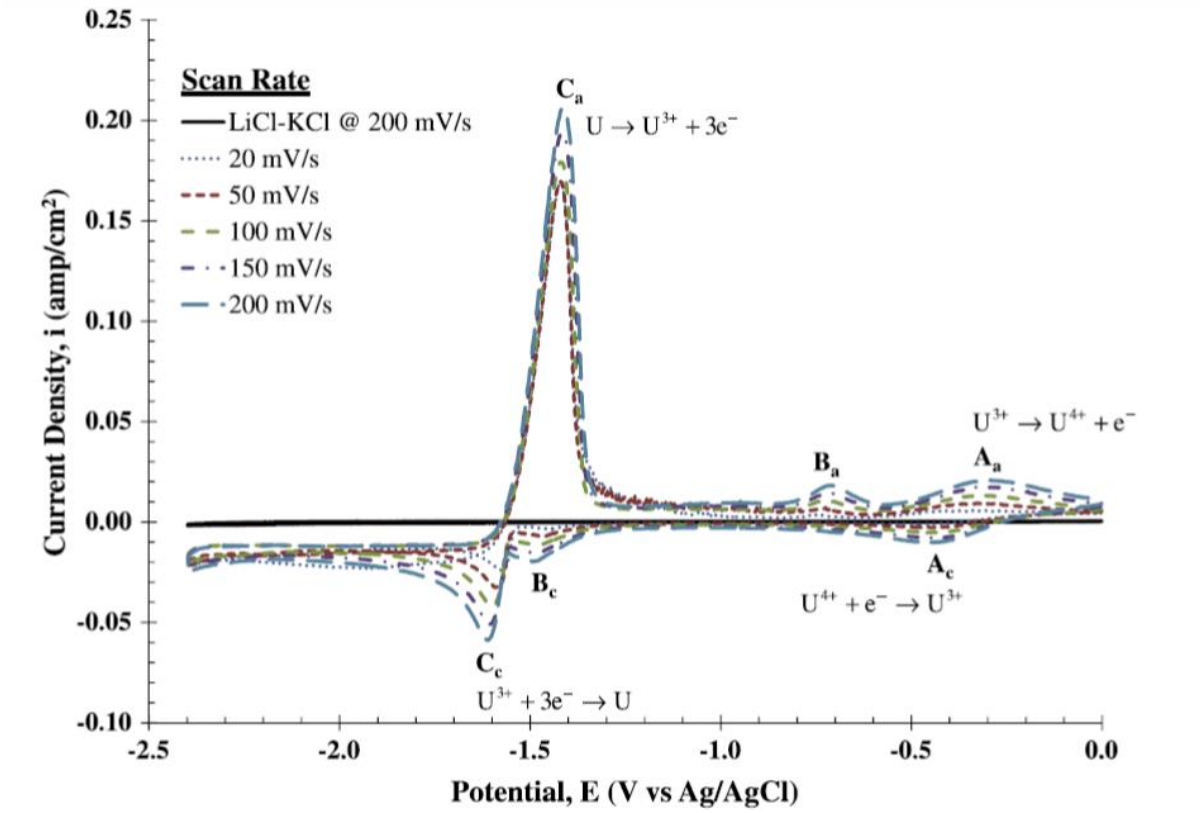

Fig 2.7 Cyclic voltammogram of 1 wt% UCl_3 in LiCL-KCl eutectic at 773K (Ref. 9).

The major cathodic and anodic peaks (C_c and C_a) show a shift of peak potential in the negative and positive direction, respectively, with respect to the scan rates (See Fig. 2.7). However, the A_c and A_a doesn't show any shift. These behaviors display that the reactions at C peaks can be considered as irreversible while those at A peaks are represented as reversible.

Chapter 3: Diffusion Model and Simulation

3.1 Analysis of Diffusion Coefficient and Apparent Standard Potential

The diffusion coefficient (D) and apparent standard potential (E^{o*}) of ions can be calculated at anodic and cathodic peak. For this purpose, experimental data sets of 1, 2.5, 5, 7.5, and 10 wt% UCl_3 and 1.07, 2.48, and 4.98 wt% $ZrCl_4$ in LiCl-KCl eutectic salt at different scan rates operating at 773 K have been considered. All raw data sets are stored in one Excel file in the way that each sheet is related to a specific concentration and scan rate. A code written in *Matlab* is able to read all sheets one by one and draws the graph related to each experiment. The user needs to select the peak by clicking on the plot and the code reports the average of diffusion coefficients and apparent standard potentials (see Appendix I).

The diffusion coefficient can be determined using the Randles-Sevcik equation for the reversible side and the Delahay equation for the irreversible side (Ref. 9, 26 and 27)

Randles- Sevcik Equation: $$D = \frac{I_{PC}^2 RT}{(0.446nFAC)^2 \upsilon nF\alpha} \quad (3.1)$$

Delahay Equation: $$D = \frac{I_{PC}^2 RT}{(0.496nFAC)^2 \upsilon nF\alpha} \quad (3.2)$$

where A is the working electrode surface area (cm^2), C is the initial concentration of active species (mol/cm^3), D is the diffusion coefficient (cm^2/s), I_{PC} is the current of cathodic peak (Amp), n is the

number of electron transferred per mole (eq/mol), F is the Faraday's constant (96485 C/eq), R is the universal gas constant (8.314 J/mol·K), T is the temperature (K), υ is the scan rate (V/s), and α is the transfer coefficient setting at 0.5.

There is a controversial issue related to utilizing transfer coefficient in Delahay equation. Based on the literatures (Ref. 28, and 29), α is combined at the reversible equation and it is not necessary to put it directly into the equation. Therefore, Eq. (3.1) in this study has been considered without α.

Initial concentration for UCl_3 in Eqs. (3.1) and (3.2) can be computed by,

$$C_i = \frac{w_i \rho}{M_i} \tag{3.3}$$

where w_i is the mass fraction of component (weight percent), C_i is the concentration (mol/cm^3), ρ is the density of LiCl-KCl (g/cm^3), and M_i is the molar mass of component i.

To calculate the density of solution in Eq. (3.3), the average density of LiCl and KCl has been estimated by following equation: (Ref. 30)

$$\rho(t) = \rho_m - k(t - t_m) \tag{3.4}$$

where ρ_m is the liquid density at the melting point (g/cm^3), t_m is the melting point (°C).

Equation (3.4) cannot be used to extrapolate more than 20°C beyond the melting point. Table 3.1 provides the values of ρ_m and t_m for LiCl and KCl. All densities of molten elements and representative salts for other materials are reported in Appendix II (Ref. 30).

Table 3.1 Molten salt data for LiCl and KCl (Ref. 30).

Formula	t_m (°C)	ρ_m (g/cm^3)	k (gr/cm^3·°C)
LiCl	610	1.502	0.000432
KCl	771	1.527	0.000583

The apparent standard potential can be related to the diffusion coefficient for a soluble/insoluble irreversible redox couple by the following equation (Ref. 27):

$$E^{0*} = E_{PC} + \frac{RT}{n\alpha F}\left[0.78 - \ln k_S + \ln\left(\sqrt{\frac{n\alpha F\nu D}{RT}}\right)\right] \quad (3.5)$$

where E_{PC} is the peak cathodic potential, E^{0*} is the apparent standard reduction potential (V), k_s is the standard rate constant (cm/s) setting at 0.00026.

Diffusion coefficient of U^{+3}/U calculated for this work is $1.101\times10^{-5} \pm 0.534\times10^{-5}$ cm^2/s; this value is close to value reported by Hoover using the CP method calculation, which is $1.04 \times 10^{-5} \pm 0.173 \times10^{-5}$ cm^2/s. This calculation is based on the average values for all different concentrations and scan rates. This is a reasonable value based on reported values in literature, ranging from 6.86×10^{-6} to 1.0×10^{-4} cm^2/s (Ref. 6, and 9, 31, 32). In addition, the average of diffusion coefficient value for U^{+4}/U^{+3} is almost 4.89×10^{-6} cm^2/s. However, Hoover (Ref. 6) reported this value around 1.26×10^{-6} cm^2/s and 6.72×10^{-6} cm^2/s via both CV and CP methods, respectively. Based on the literature survey, the diffusion coefficient of U^{+4}/U^{+3} can be ranging between 7.29×10^{-6} to 2.73×10^{-5} cm^2/s (Ref. 6, 9, 28, and 33). It should be noted that more D and E^{0*} values reported in the literatures for uranium chloride are collected and listed in Appendix III.

3.2 Numerical Method and Approach of Diffusion Model[1]

3.2.1 Fundamental Equations

Two general forms of current as a function of time for reversible and irreversible conditions can be utilized to compute the current based on increasing time; these are (Ref. 34):

$$i_{reversible} = nFAC_o^* \sqrt{\pi D_o \psi} \chi(\psi t) \tag{3.6}$$

$$i_{irreversible} = nFAC_R^* \sqrt{\pi D_R \alpha \psi} \chi(\alpha \psi t) \tag{3.7}$$

where $\psi = \frac{nF\upsilon}{RT} = (\frac{nF}{RT})(E_i - E)$, C_o^* and C_R^* are the bulk concentration of oxidant and reductant species (mol/cm^3), respectively, D_o and D_R are the diffusion coefficients of oxidant and reductant species (cm^2/s), respectively, and E_i is the initial potential (V). In general, the average values of D_o and D_R would be determined experimentally via both CV and CP methods. In addition, the apparent standard potentials of uranium were reported by several researchers [Refs. 26, 27, 33 to 41]. Therefore, in this study, the reported values of the diffusion coefficient and apparent standard potentials were utilized for an initial estimation of diffusion coefficients and formal electrode potentials (Ref. 6). This implies that after providing the χ function, current distribution at different times can be computed through Eqs. (3.6) and (3.7).

[1] Content in this Chapter are cited from the author's publication:
Samaneh Rakhshan Pouri, Supathorn Phongikaroon, "An Interactive Reverse-Engineering Cyclic Voltammetry for Uranium Electrochemical Studies in LiCl-KCl Eutectic Salt", *Journal of Nuclear Technology*, Vol. 197, No. 3, pp.308-319 (2017).

3.2.2 Current and Potential for the Reversible Part

Generally, Fick's law can be applied to each species (Ref. 4). By applying Fick's law on a reversible electron reaction of $O^{n+} + ne^- \Leftrightarrow R$ where O and R are related to oxidized and reduced species, respectively, two partial differential equations can be given by

$$\frac{\partial C_o(x,t)}{\partial t} = D_o \frac{\partial^2 C_o(x,t)}{\partial x^2} \tag{3.8}$$

$$\frac{\partial C_R(x,t)}{\partial t} = D_R \frac{\partial^2 C_R(x,t)}{\partial x^2} \tag{3.9}$$

where t is the time (s), and x is the linear distance from the electrode surface (cm). Initial conditions for both differential equations ($x \geq 0$) are

$$C_o(x,0) = C_o^*(x,0) \tag{3.10}$$

$$C_R(x,0) = 0 \tag{3.11}$$

Here, two boundaries conditions for $t > 0$ are given by

$$D_o \frac{\partial C_o(0,t)}{\partial x} = -D_R \frac{\partial C_R(0,t)}{\partial x} \tag{3.12}$$

$$C_o(0,t) = C_R(0,t) \exp[(\frac{nF}{RT})(E - E^\circ)] \tag{3.13}$$

where E° is the formal electrode potential (V). The potential at Nernst equation, Eq. (3.13), can be defined within two regions: (1) from the initial potential toward further negative direction and (2) from the reversed potential toward the positive direction; these are, respectively,

$$E = E_i - \upsilon t \qquad 0 < t \leq \varepsilon \tag{3.14}$$

$$E = E_i - 2\varepsilon\upsilon + \upsilon t \qquad t \geq \varepsilon \tag{3.15}$$

where ε is the time (s) that the potential scan is reversed toward the positive potential and anodic scanning is started. Substituting Eqs. (3.14) or (3.15) into Eq. (3.13) yields this simplified form of:

$$\frac{C_o}{C_R} = \phi_{rev} S_\varepsilon(t) \tag{3.16}$$

where

$$\phi_{rev} = \exp[(\frac{nF}{RT})(E_i - E^\circ)] \tag{3.17}$$

and

$$S_\varepsilon(t) = \begin{cases} e^{-\psi t} & t \leq \varepsilon \\ e^{\psi t - 2\upsilon\varepsilon} & t \geq \varepsilon \end{cases}. \tag{3.18}$$

The following integral solutions for oxidant and reductant surface concentrations after applying the Laplace transform and convolution theorem are (Refs. 4, 42 and 43):

$$C_o(0,t) = C_o^*(x,0) - \frac{1}{nFA\sqrt{\pi D_o}} \int_0^t \frac{i(\tau)d\tau}{\sqrt{t-\tau}} \tag{3.19}$$

$$C_R(0,t) = \frac{1}{nFA\sqrt{\pi D_R}} \int_0^t \frac{i(\tau)d\tau}{\sqrt{t-\tau}} \tag{3.20}$$

Dividing Eq. (3.19) by Eq. (3.20) and substituting it into Eq. (3.16), the integral part can be further expressed as:

$$\int_0^t \frac{i(\tau)d\tau}{\sqrt{t-\tau}} = \frac{nFAC_o^*\sqrt{\pi D_o}}{1 + \gamma\phi_{rev} S_\varepsilon(t)} \tag{3.21}$$

where $\gamma = \sqrt{\frac{D_o}{D_R}}$. Since Faradaic current can be described by

$$f(t) = D_o\left(\frac{\partial C_o}{\partial x}\right)_{x=0} = \frac{i_{faradaic}}{nFA}. \tag{3.22}$$

Thus, by substituting Eq. (3.22) into Eq. (3.21), the left-hand-side (LHS) integral of Eq. (3.21) becomes

$$\int_0^t \frac{f(\tau)d\tau}{\sqrt{t-\tau}} = \frac{C_o^*\sqrt{\pi D_o}}{1+\gamma\phi_{rev}S_\varepsilon(t)}. \tag{3.23}$$

To provide a dimensionless integral, $f(\psi t) = C_o^*\sqrt{\pi D_o}\chi(\psi t)$ can be substituted in Eq. (3.23), which can be expressed as:

$$\int_0^{\psi t} \frac{\chi(\psi t)d(\psi t)}{\sqrt{\psi t-\psi\tau}} = \frac{1}{1+\gamma\phi_{rev}S_{\psi\varepsilon}(\psi t)}. \tag{3.24}$$

It is more convenient to change the variable as $\psi\tau = \delta\nu$ and $\psi t = \delta j$ and Eq. (3.24) can be expressed as:

$$\sqrt{\delta}\int_0^{\delta j} \frac{\chi(\delta\nu)d(\nu)}{\sqrt{j-\nu}} = \frac{1}{1+\gamma\phi_{rev}S_{\delta\varepsilon}(\delta j)} \tag{3.25}$$

where j is a serial number of the subinterval in integral steps, and δ is the length of subinterval. After applying integration by parts and Riemann-Stieltjes integral on the LHS of Eq. (3.25), the final result becomes (Refs. 4, 29, 44, and 45):

$$2\sqrt{\sigma}\left[\chi(1)\sqrt{j} + \sum_{i=1}^{j-1}\sqrt{j-i}[\chi(i+1)-\chi(i)]\right] = \frac{1}{1+\gamma\phi_{rev}S_{\delta\varepsilon}(\delta j)} \tag{3.26}$$

It should be noted that Eq. (3.26) provides the j equations with j-1 unknowns function $\chi(j)$ of the previous step. These equations can be solved for values of $\chi(\delta j)$, and give the current at each time based on Eqs. (3.6) or (3.7). The potential at reversible part can be calculated by using Eqs. (3.13) and (3.16) and getting natural logarithm as follow:

$$E = E^{\circ} + \frac{RT}{nF}[\ln \phi_{rev} + \ln S_{\psi\varepsilon}(\psi t)] . \tag{3.27}$$

3.2.3 Current and Potential for the Irreversible Part

The initial conditions for an irreversible equation, $R \Rightarrow O^{n+} + ne^{-}$, has the same initial condition as that for the reversible equations (Eqs. (3.10) and (3.11)). However, the irreversible equation has an additional boundary condition for $t > 0, x = 0$, which is

$$k_S \exp\left[(\frac{-\alpha nF}{RT})(E - E^{\circ})\right] C_R = D_R(\frac{\partial C_R}{\partial x}) = C_R k_i e^{bt} \tag{3.28}$$

where k_S is the standard rate constant and is considered as 0.00026 (Ref. 27), $b = \alpha\psi$, and k_i is defined as

$$k_i = k_S \exp[(\frac{-\alpha nF}{RT})(E_i - E^{\circ})] . \tag{3.29}$$

There is an assumption at the irreversible part to ease the calculations quickly by using the current reversible equation for the irreversible part without applying the new boundary conditions of Eq. (3.28). For making the irreversible more realistic, this condition (Eq. (3.28)) has been used for potential calculations. Using a similar approach as discussed in Section II.B, the integral provided by Delahay for irreversible condition can be expressed as

$$1-\int_{0}^{bt}\frac{\chi(\delta\nu)d(\nu)}{\sqrt{bt-\nu}}=e^{u-bt}\chi(bt) \tag{3.30}$$

where

$$e^{u}=\frac{\sqrt{\pi D_{o}b}}{k_{i}}. \tag{3.31}$$

After substituting Eq. (3.31) into Eq. (3.29) and applying the natural logarithm, the potential of irreversible part can be calculated by:

$$E=E^{o}+(\frac{RT}{\alpha nF})(u-bt)-(\frac{RT}{\alpha nF})\ln(\frac{\sqrt{\pi D_{R}b}}{k_{S}}). \tag{3.32}$$

3.2.4 Surface Concentration for both Reversible and Irreversible Parts

The cyclic voltammetry plot is divided into four major regions: (1) the reversible cathodic, (2) the irreversible cathodic, (3) irreversible anodic, and (4) reversible anodic. For calculating the surface concentration of oxidant species at the reversible part, Eq. (3.21) is substituting into Eq. (3.19) to remove the integral part yielding:

$$C_{o}=C_{o}^{*}(\frac{\gamma\phi_{rev}S_{\varepsilon}(t)}{1+\gamma\phi_{rev}S_{\varepsilon}(t)}) \tag{3.33}$$

Here, the initial concentration of reduction at the beginning is zero (Eq. (3.11)). Therefore, based on the conservation of mass, the concentration of reduction species at reversible part can be calculated by using the fact that $C_{R}=C_{o}^{*}-C_{o}$. In addition, concentrations for the irreversible side can be defined by:

$$C_o(0,t) = \frac{1}{nFA\sqrt{\pi D_o}} \int_0^t \frac{i(\tau)d\tau}{\sqrt{t-\tau}} \tag{3.34}$$

$$C_R(0,t) = C_R^*(x,0) - \frac{1}{nFA\sqrt{\pi D_R}} \int_0^t \frac{i(\tau)d\tau}{\sqrt{t-\tau}} \tag{3.35}$$

After dividing Eq. (3.34) by Eq. (3.35) and substituting it into Eq. (3.16), the integral part can be expressed as:

$$\int_0^t \frac{i(\tau)d\tau}{\sqrt{t-\tau}} = \frac{nFAC_R^*\phi_{irrev}S_\varepsilon(t)\sqrt{\pi D_o}}{1+\gamma\phi_{irrev}S_\varepsilon(t)} \tag{3.36}$$

where $\phi_{irrev} = \exp[(\frac{n\alpha F}{RT})(E_i - E^\circ)]$. Therefore, the concentration of oxidant species for the irreversible parts can be expressed as

$$C_R = C_R^*\left(\frac{1}{1+\gamma\phi_{irrev}S_\varepsilon(t)}\right). \tag{3.37}$$

Again, at this range, the initial concentration of oxidant species are negligible and $C_o = C_R^* - C_R$.

It is assumed that the initial concentration of U^{+3} at the cathodic irreversible part is the initial concentration of UCl_3 at the bulk. Therefore, the concentration calculation is started from irreversible cathode part due to the fact that the initial concentration of U^{+4} is unknown at the reversible cathodic side. Concentration of oxidant and reductant species can be calculated through several modes:

Mode 1: Irreversible cathode:

$$C_{oCirrev} = C_o^*(1 - \frac{1}{1+\gamma\phi_{irrev}S_\varepsilon(t)}) \tag{3.38}$$

where $C_{oCirrev}$ is the concentration of oxidant species at the cathodic irreversible side (mol/cm^3).

Mode 2: Irreversible Anode:

$$C_{RAirrev} = C_{RCirrev}(1 - \frac{\gamma\phi_{ireev}S_{\varepsilon}(t)}{1+\gamma\phi_{ireev}S_{\varepsilon}(t)}) \tag{3.39}$$

where $C_{RAirrev}$ and $C_{RCirrev}$ are the concentrations of reductant species at anodic and cathodic irreversible sides (mol/cm^3), respectively. It should be noted that $C_{RCirrev} = C_o^* - C_{oCirrev}$.

Mode 3: Reversible Anode:

$$C_{RArev} = C_{oAirrev}(1 - \frac{\gamma\phi_{rev}S_{\varepsilon}(t)}{1+\gamma\phi_{rev}S_{\varepsilon}(t)}) \tag{3.40}$$

where C_{RArev} is the concentration of reductant species at the anodic reversible side (mol/cm^3) and $C_{oAirrev}$ is the concentration of oxidant species at the anodic irreversible side (mol/cm^3) with the fact that $C_{oAirrev} = C_{RCirrev} - C_{RAirrev}$.

Mode 4: Reversible Cathode:

$$C_{oCrev} = C_{oArev}(1 - \frac{1}{1+\gamma\phi_{rev}S_{\varepsilon}(t)}) \tag{3.41}$$

where C_{oCrev} and C_{oArev} are the concentrations of oxidant species at the cathodic and anodic reversible sides (mol/cm^3), respectively. Here, $C_{oArev} = C_{oAirrev} - C_{RArev}$ and $C_{RCrev} = C_{oArev} - C_{oCrev}$ where C_{RCrev} is the concentration of reductant species at the reversible cathodic side (mol/cm^3).

3.3 Uranium Chloride

3.3.1 Computational Procedure

The numerical technique was implemented within a commercial software package *Matlab* (R2017a) through the Windows PC computer with the following configurations: Intel Core i5, 3.3 GHZ, 16 GB, and 1 TB. The simulation was related to an electrochemical study of UCl_3 in LiCl-KCl eutectic salt at 1, 2.5, 5, 7.5, and 10 wt% UCl_3 concentrations under different scan rates operating at 773 K (Ref. 6). It is important to mention that all experimental runs were conducted and reported by Hoover (Ref. 6). Detailed experimental setup and conditions can be found in both Refs. 6 and 9. The CV measurements were performed with different concentrations of UCl_3 in LiCl-KCl salt at various scan rates: 1 wt% and 2.5 wt% at 100 mV/s to 300 mV/s; 5 wt% at 400 mV/s to 2000 mV/s; 7.5 wt% at 400 mV/s to 1500 mV/s, and 10 wt% at 200 mV/s to 1000 mV/s. Different chosen scan rates were based on the experimental program to determine the reversible and irreversible peaks. The shift of the major anodic and cathodic peaks was more prevalent at the higher range of concentrations. Therefore, the higher scan rates were being applied at higher concentrations to find the peaks clearly. Another reason of using the high scan rate at high concentration condition was to avoid the massive deposition that would occur on the electrode surface.

In the CV curves, four peaks were observed: P_{C1} (region 1), P_{C2} (region 2), P_{a2} (region 3), and P_{a1} (region 4) where subscripts C and a stand for cathode and anode, respectively (these peaks will be shown later in the CV plot). The run time for this code at each concentration and scan rate takes less than two minutes when the time interval (δ) is around 0.08 seconds. However, if δ

decreases to 0.024 seconds, the simulation time increases significantly. For example, the complete processing time (providing Current Vs. Potential and Concentrations Vs. Time graphs) for 1 wt% with 100 mV/s by using $\delta = 0.08$ s would take approximately 95 s while the processing time would require up to 3515 s by using $\delta = 0.024$ s. This result implies that decreasing δ by 70% may cause the processing time to increase by 3600%.

As mentioned above, Ref. 6 provides the average of diffusion coefficients and apparent standard potentials and these reported data sets were utilized to determine the diffusion coefficients and formal electrode potentials in this work. For this purpose, the diffusion coefficients and formal electrode potential was tuned with 10^{-7} and 0.0002 interval, respectively. This step needs few iterations to improve the fit with experimental values. For example, if the cathodic irreversible theoretical peak was below the experimental peak, then the diffusion coefficient would be increased to justify that. If the potential of cathodic irreversible theoretical peak is on the left side of the experimental peak, the formal potential should be increased to adjust the peak to the right. Using these input data sets (diffusion coefficients, formal electrode potentials, and the process time), it is possible to predict the trend of CV graph based on the given information; all these values are listed in Table 3.3. As mentioned above, the cyclic voltammetry is divided into four major regions (regions 1 to 4). The scanning times for each region are listed in the bottom part of Table 3.3. It should be noted that the initial time is the starting time of currents and potentials for each recorded run.

In addition to the written *Matlab* code, this work was conformed in a GUI environment. The GUI layout and output example are illustrated in Figs. 3.1 and 3.2. All the data which are used at the codes are reported in Table 3.2, and 3.3. These information was stored in the GUI code so

the user can choose the concentration of uranium and the scan rate, then the current (amp) versus potential (V) graph and concentration (g/cm^3) of each species over the time (s) can be plotted. The *Matlab* and GUI codes for diffusion model concentration graph and CV plot can be found in Appendix IV.

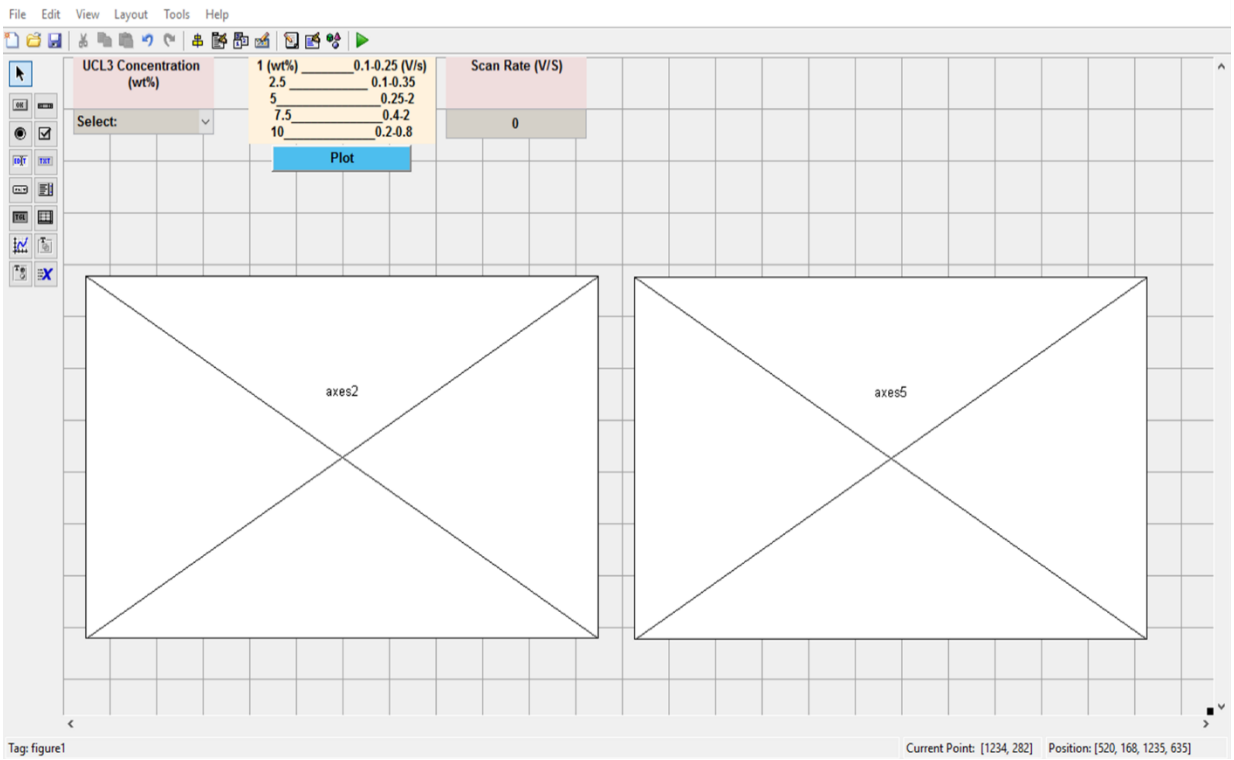

Fig. 3.1 The GUI Layout Editor.

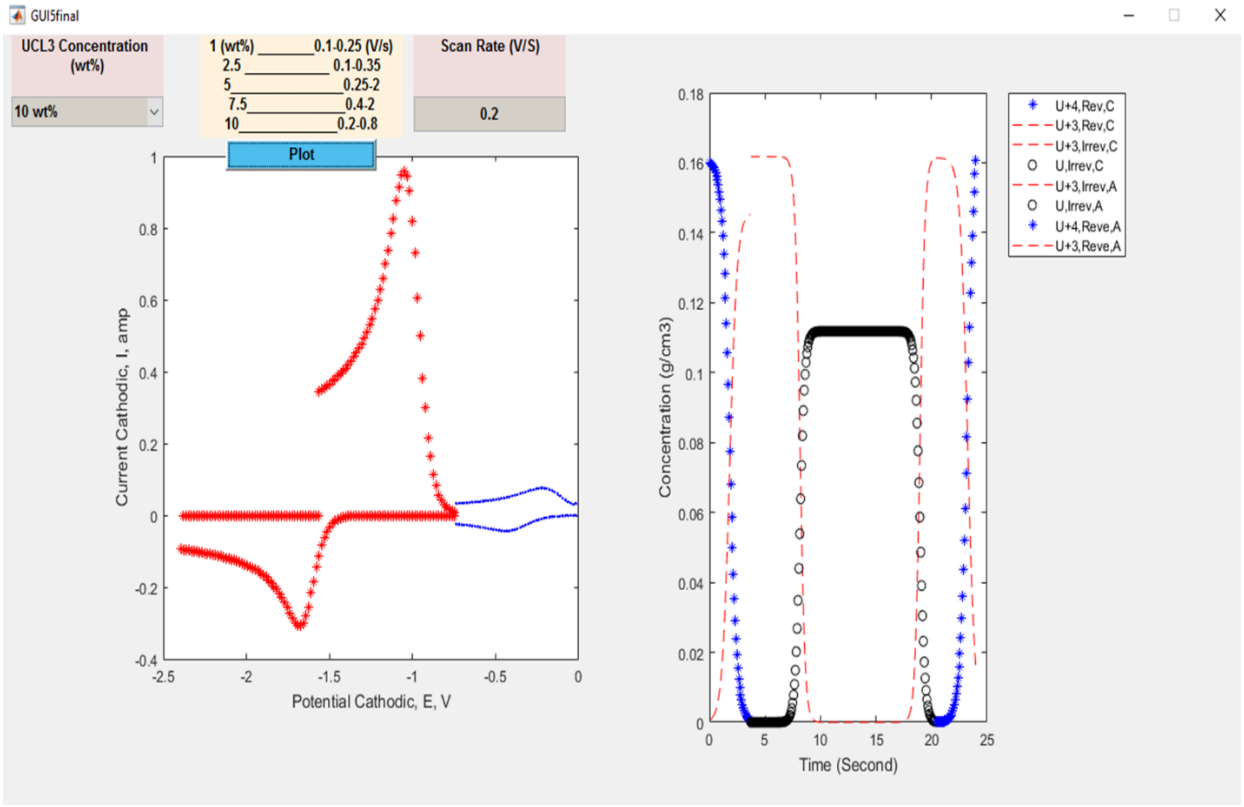

Fig. 3.2 GUI output for 10 wt% UCl_3 in LiCl-KCl eutectic at 773 K with a scan rate of 200 mV/s.

Table 3.2 Diffusion coefficient and formal electrode potentials for uranium 1, 2.5, and 5 wt% at different scan rates.

Wt %, Scan Rate (mV/s)	1, 100	1, 150	1, 200	2.5, 100	2.5, 150	2.5, 200	2.5, 300	5, 400	5, 600	5, 900	5, 1000
$D(U^{+4}/U^{+3})$ $cm^2/s\times10^5$	1.05	1.05	1.05	0.70	0.70	0.70	1.05	0.45	0.45	0.45	0.45
$D(U^{+3}/U^{+4})$ $cm^2/s\times10^5$	3.02	3.02	3.02	3.02	2.70	3.02	3.02	2.40	2.40	2.40	2.40
E^0 cathode (V)	-0.43	-0.43	-0.43	-0.43	-0.43	-0.43	-0.43	-0.43	-0.43	-0.43	-0.43
E^0 Anodic (V)	-0.25	-0.25	-0.25	-0.25	-0.25	-0.25	-0.25	-0.25	-0.25	-0.23	-0.23
$D(U^{+3}/U)$ $cm^2/s\times10^5$	2.00	2.00	2.00	2.00	2.00	1.82	1.75	1.45	1.50	1.18	1.18
$D(U/U^{+3})$ $cm^2/s\times10^5$	3.80	2.90	2.50	3.00	2.15	1.65	1.30	0.85	0.70	0.53	0.52
E^0 cathode (V)	-1.61	-1.61	-1.61	-1.6	-1.60	-1.61	-1.62	-1.67	-1.67	-1.7	-1.7
E^0 Anode (V)	-1.45	-1.42	-1.41	-1.34	-1.34	-1.34	-1.32	-1.19	-1.16	-1.13	-1.13
Initial Time $(s)\times10^{-3}$	12	12	12	1.2	1.2	1.2	1.2	6	6	4	4
Reversible Cathode Time (s) (region 1)	9.36	6.62	4.76	9.74	6.67	3.99	3.16	2.48	1.64	0.96	0.81
Irreversible Cathode Time (s) (region 2)	14.54	15.83	12.08	24.05	16.00	12.11	8.00	6.01	3.98	2.66	2.39
Time Irreversible Anode (s)	14.54	25.03	19.40	38.35	25.33	20.22	12.84	9.54	6.32	4.37	3.98
Time Reversible Anode (s)	9.34	31.65	24.15	48.09	32.00	24.22	16.00	12.02	7.95	5.32	4.79
Time Interval (s)	0.08	0.08	0.08	0.08	0.08	0.08	0.08	0.08	0.04	0.04	0.04

Table 3.3 Diffusion coefficient and formal electrode potentials for uranium 7.5, and 10 wt% at different scan rates.

Wt %, Scan Rate (mV/s)	7.5, 400	7.5, 500	7.5, 1000	7.5, 1400	7.5, 2000	10, 200	10, 500	10, 800
$D(U^{+4}/U^{+3})$ $cm^2/s\times10^5$	0.45	0.49	0.5	0.5	0.5	0.25	0.20	0.13
$D(U^{+3}/U^{+4})$ $cm^2/s\times10^5$	1.85	1.85	2.2	2.2	2.45	0.75	1.50	1.8
E^0 cathode (V)	-0.43	-0.43	-0.5	-0.5	-0.5	-0.40	-0.55	-0.55
E^0 Anodic (V)	-0.20	-0.20	-0.16	-0.16	-0.145	-0.18	-0.18	-0.18
$D(U^{+3}/U)$ $cm^2/s\times10^5$	1.40	1.40	1.05	0.99	0.94	0.93	0.83	0.65
$D(U/U^{+3})$ $cm^2/s\times10^5$	0.95	0.75	0.50	0.418	0.34	0.9	0.44	0.3
E^0 cathode (V)	-1.70	-1.70	-1.75	-1.79	-1.81	-1.68	-1.73	-1.75
E^0 Anode (V)	-1.09	-1.07	-0.98	-0.94	-0.88	-1.05	-1.00	-0.95
Initial Time $(s)\times10^{-3}$	1.2	1.2	1.2	1.2	1.2	1.2	1.2	1.2
Reversible Cathode Time (s) (region 1)	1.92	1.33	0.65	0.494	0.34	3.65	1.28	0.85
Irreversible Cathode Time (s) (region 2)	5.99	4.74	2.39	1.72	1.195	11.98	4.792	2.991
Time Irreversible Anode (s)	10.06	8.16	4.13	2.95	2.051	20.94	8.33	5.14
Time Reversible Anode (s)	11.97	9.48	4.79	3.45	2.39	23.98	9.59	5.98
Time Interval (s)	0.03	0.03	0.02	0.02	0.02	0.08	0.08	0.05

3.3.2 Results

Plots showing the relationship between current (amp) and potential (V) for UCl_3 in LiCl-KCl eutectic salt at 1, 5, and 10 wt% UCl_3 concentrations under different scan rates are displayed in Figs. 3.3 to 3.5, respectively. Here, the predicted simulated results are being compared to the experimental data sets (Ref. 6). There are two distinctive colors shown in these figures superimposing on top of the experimental data displayed in black: (1) the blue color (dot) trends on the right side indicate both reversible of cathodic and anodic reactions, and (2) the red color (asterisk) trends on the left side indicate irreversible cathodic and anodic reactions.

Although the results reveal that this method can predict the cathodic and anodic peaks, the shape of predicted anodic irreversible trend is not exactly the same as that of the experimental data. This dissimilarity indicates that the Fick's law can only provide a proper outcome for a Gaussian trend. In addition, the results indicate that this method does not predict the adsorption peaks in both anodic and cathodic regions. Also, the plots for high concentrations (10 wt%) of the irreversible cathodic side have some unpredicted parts in comparing to the low concentrations (Fig. 3.5). But the main focus of this study is related to the peaks in the absence of experimental data in order to provide us such information (e.g., the current and potential). Overall, the results capture the important features of the CV graph such as the potential and current information at each peak with a small error. For example, the average root mean square error (RMS) of potential and current for 1 wt% with 100 mV/s are 0.00764 and 0.0178, respectively. The calculating results also indicate that when concentration increases, the average RMS increases. Here, the average RMS of potential and current for 5 wt% with 400 mV/s are 0.09116 and 0.0632, respectively. The

results also indicate that Delahay and Randles–Sevcik equations can also be used to predict high concentration conditions.

In the case that the input information at a specific scan rate is not accessible, the code interpolates linearly between two available input sets, which are stored at GUI code, such as diffusion coefficient, formal electrode potential, and time duration. Afterward, the code calculates the current, potential and concentration over time by using the interpolated values. Fig. 3.6 illustrates the simulated results of 2.5 wt% at 200 mV/s through an interpolation between 150 mV/s and 300 mV/s of 2.5 wt% data sets. The differences between the interpolation method and Hoover's work of the current and potential peaks are approximately 6.2% and 11.1%, respectively.

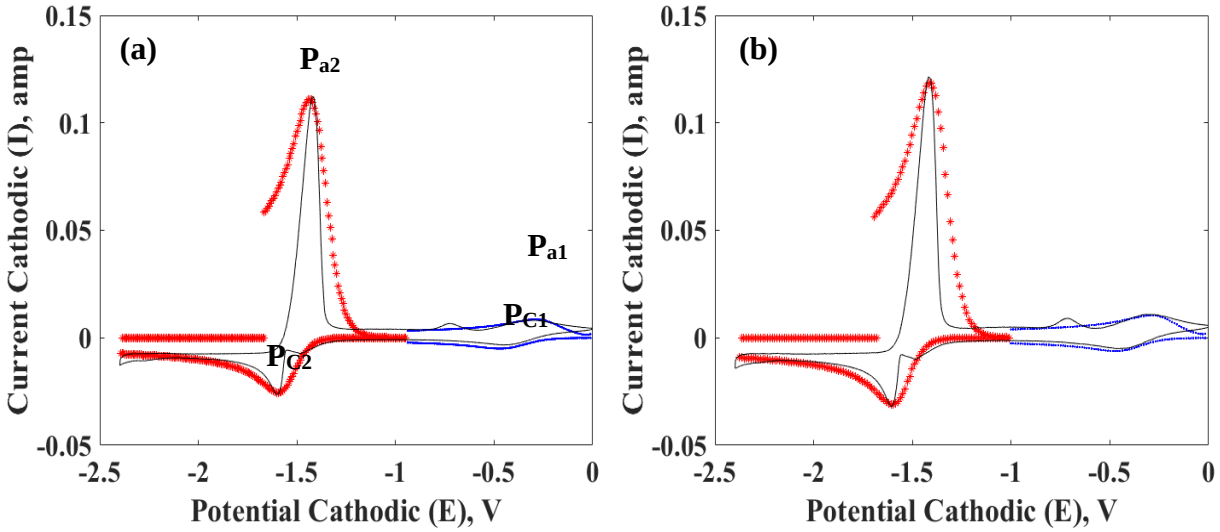

Fig. 3.3 Cyclic voltammograms of 1 wt% UCl_3 in LiCl-KCl eutectic at 773 K with a working electrode surface area of 0.626 cm^2 at the scan rate of (a) 100 mV/s and (b) 150 mV/s.

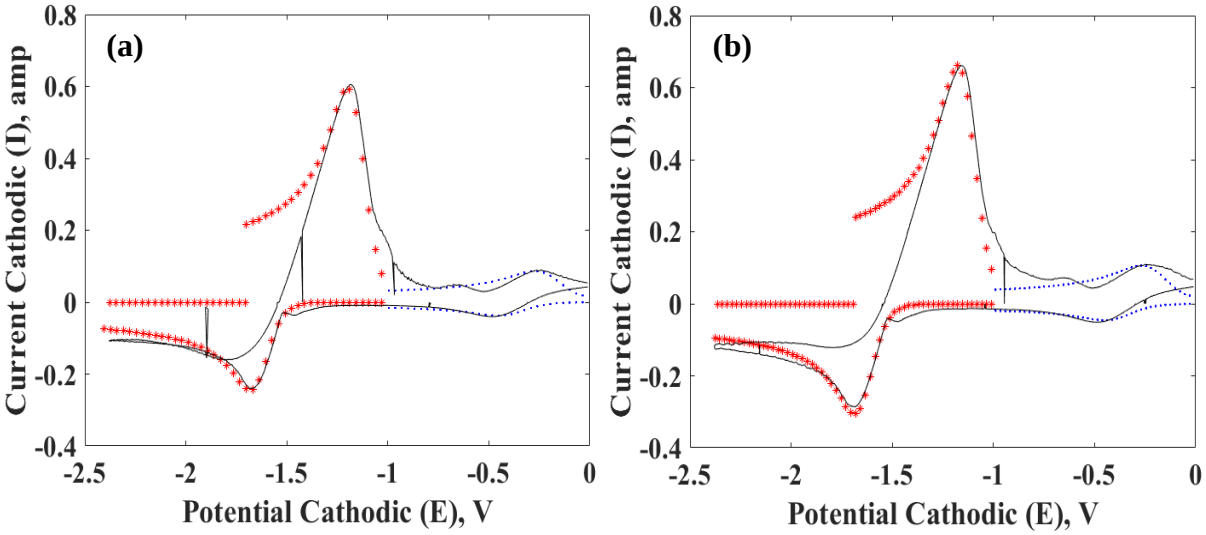

Fig. 3.4 Cyclic voltammograms of 5 wt% UCl_3 in LiCl-KCl eutectic at 773 K with a working electrode surface area of 0.710 cm^2 at the scan rate of (a) 400 mV/s and (b) 600 mV/s.

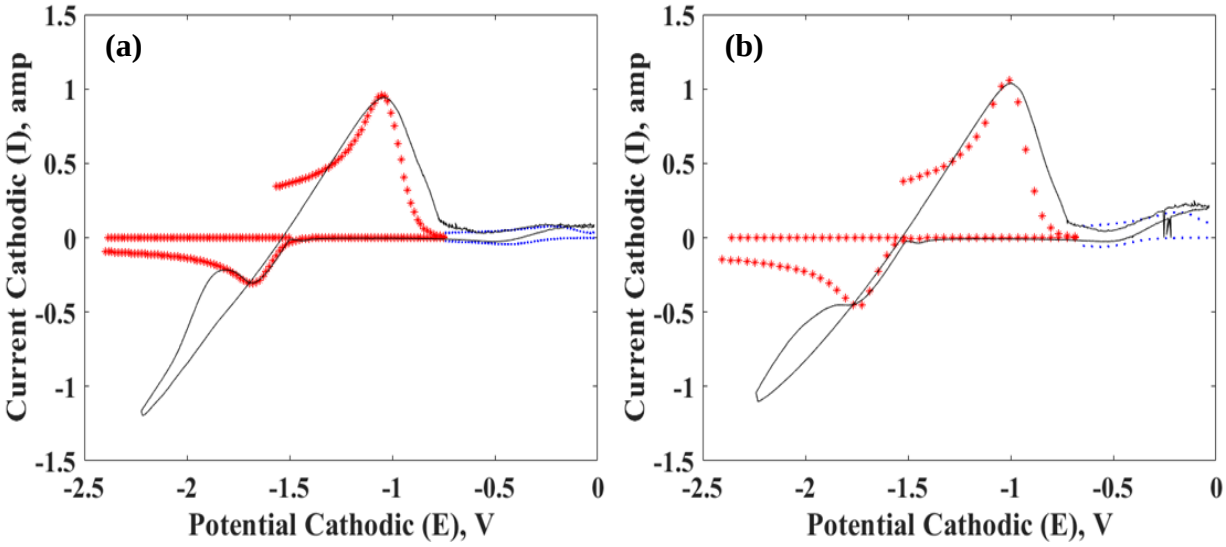

Fig. 3.5 Cyclic voltammograms of 10 wt% UCl_3 in LiCl-KCl eutectic at 773 K with a working electrode surface area 0.785 cm^2 at the scan rate of (a) 200 mV/s and (b) 500 mV/s.

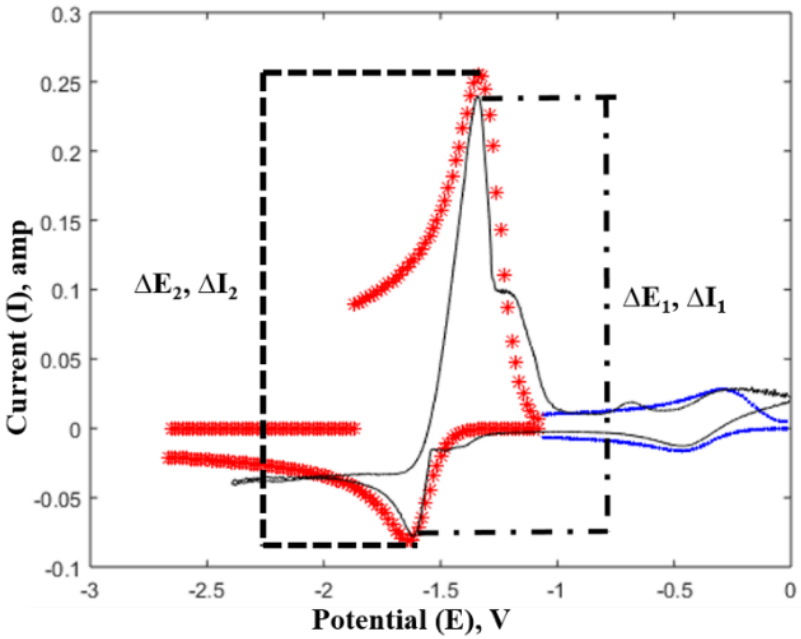

Fig. 3.6 Cyclic voltammograms of 2.5 wt% UCl_3 in LiCl-KCl eutectic at 773 K at the scan rate of 200 mV/s by interpolating input data between 150 and 300 mV/s.

In addition to the CV tracing, this method can be used to calculate the concentration of each species with different initial bulk concentrations at various scan rates. Fig. 3.7 shows the concentrations of reduced and oxidized species for 1 and 10 wt% UCl_3 in LiCl-KCl at 100 and 200 mV/s, respectively. Similar trends can be observed at other concentrations and scan rates. Fig. 3.7 illustrates that the concentration of U^{+3} at the irreversible cathodic side decreases and the concentration of uranium metal at the electrode surface increases. When the scan reverses, the concentration of uranium reduces and a concentration of U^{+3} grows at the irreversible anodic side. This concentration declines at the reversible anodic side. Then the U^{+4} concentration goes up at the irreversible anodic side and goes down at the reversible cathodic side within 10 s time frame. It is essential to note that the concentration of uranium chloride at the reversible cathodic and anodic sides are slightly different due to absorption of U^{+3} at the cathodic part. In addition, the

reason that uranium concentration is lower than uranium chloride is due to losing the chloride and depositing as the uranium metal at the electrode surface. The results at any desired concentrations (up to 10 wt% UCl_3) can be displayed through the GUI environment (see Fig. 3.2). Through *Matlab*, the user is able to calculate the reduction and oxidation concentration of each point. The number of desired points can be entered by a user in the code to deliver the concentration of reduction and oxidation species, and the process time at each point (as illustrated in Table 3.4). An example of this capability for 1 wt% uranium at 100 mV/s with four selected points is shown in Fig. 3.8. This routine can be accomplished within one minute.

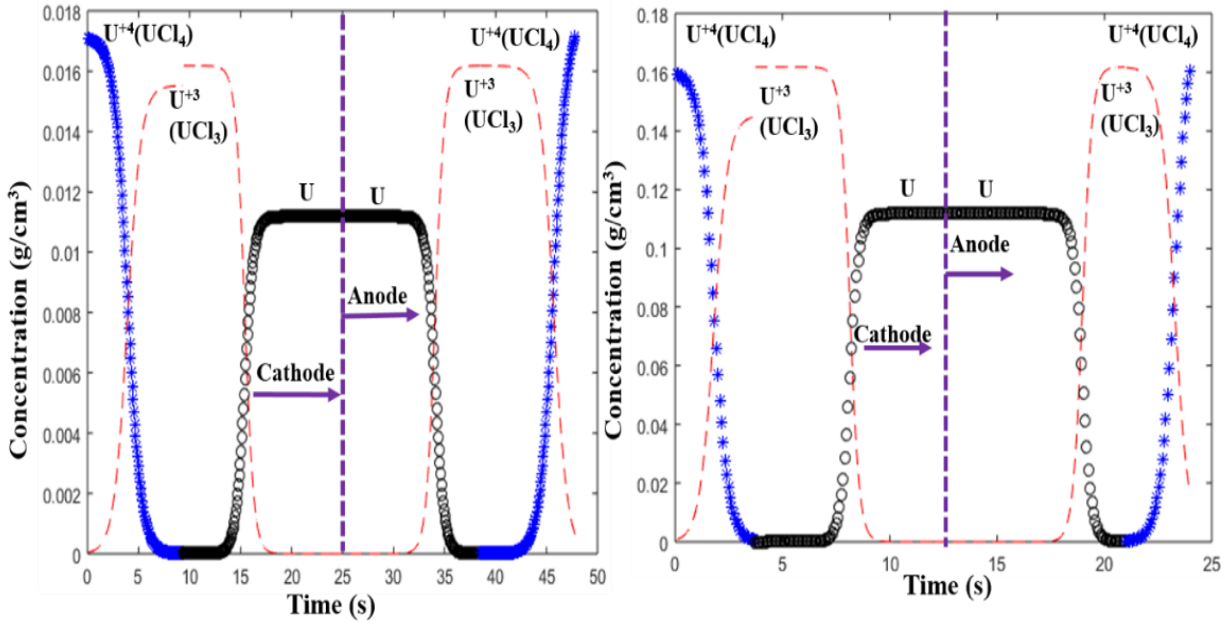

Fig. 3.7 The concentration of Reduced and Oxidized species for 1wt% UCl_3 in LiCl-KCl eutectic at 773 K with 100 mV/s (left) and 10 wt% at 200 mV/s (right).

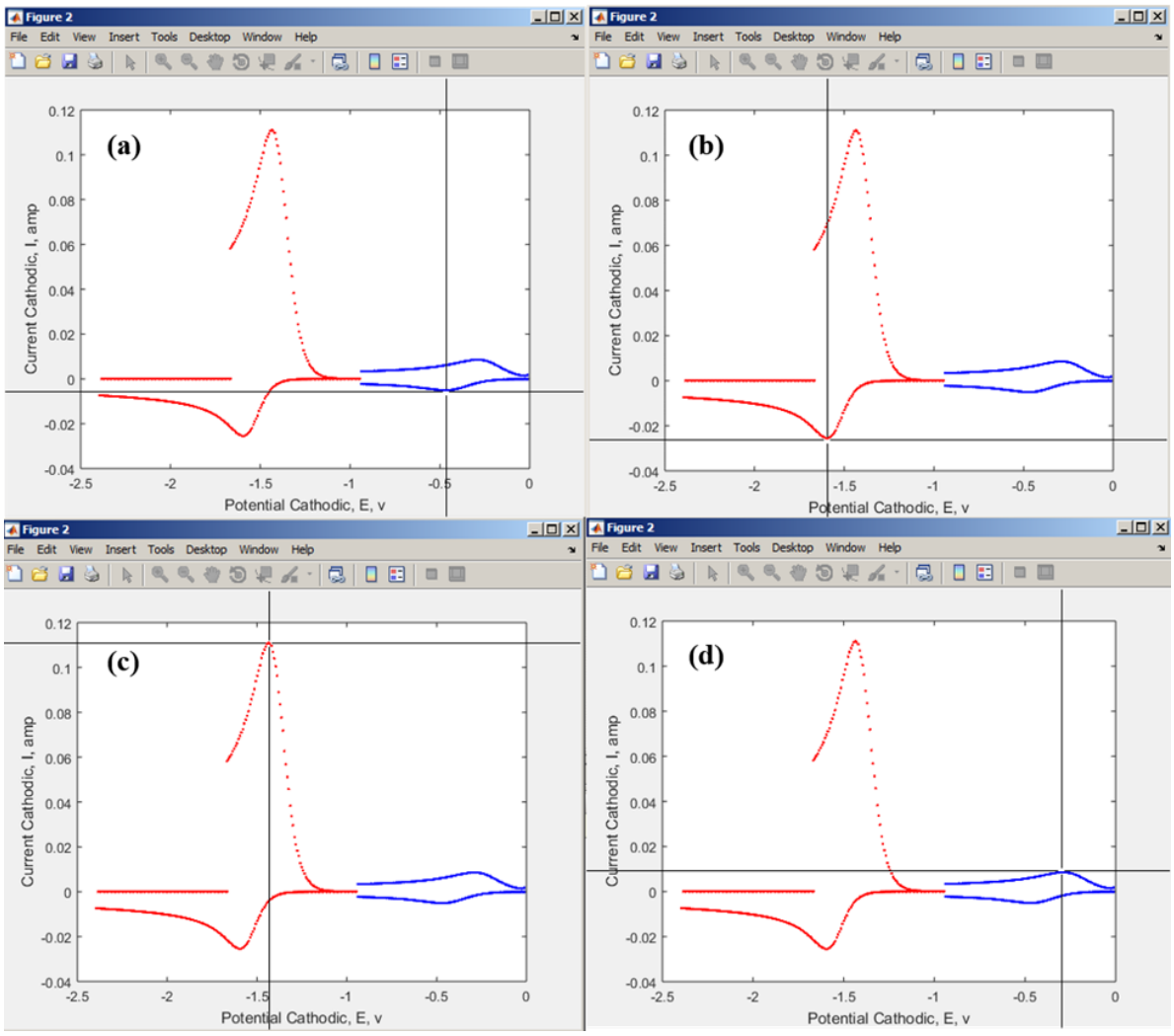

Fig. 3.8 Selected points for calculating the concentration of 1 wt% at 100 mV/s for (a) cathodic reversible, (b) cathodic irreversible, (c) anodic irreversible, and (d) anodic reversible.

Table 3.4 Oxidation and reduction concentrations, and the process time of the selected points related to Fig. 3.8.

Points	Time (Second)	Reduction Concentration		Oxidation Concentration	
		(g/cm^3)	(mol/cm^3)	(g/cm^3)	(mol/cm^3)
a	4.55	0.01120	3.252×10^{-5}	0.00477	1.256×10^{-5}
b	15.89	0.00838	3.520×10^{-5}	0.00405	1.176×10^{-6}
c	33.58	0.00852	3.579×10^{-5}	0.00385	1.118×10^{-5}
d	44.54	0.01377	3.998×10^{-5}	0.00266	7.003×10^{-6}

3.4 Zirconium Chloride

3.4.1 Computational Procedure

Cyclic voltammogram of zirconium chloride is complex in nature due to several mechanistic reactions occurring during each CV run. Therefore, the developed model experienced difficulty in fitting experimental data sets due to two following reasons: (1) there are some controversial issues related to the reactions happening at each peak, and (2) the diffusion coefficients at each peak are not clear or existed in literatures. The reactions existence at each peak based on the literatures (Refs. 26, 35, and 36) and Hoover's study (Ref. 6 and 9) are illustrated in Fig. 3.9, indicating that (i) the potential is being scanned from 0.5 V in negative direction at cathodic side and (ii) the first cathodic peak shows Zr^{+4}/Zr^{+2} or Zr^{+2}/Zr reduction or combination of both while the second and third peaks are for Zr^{+2}/Zr and Zr^{+4}/Zr reductions, respectively.

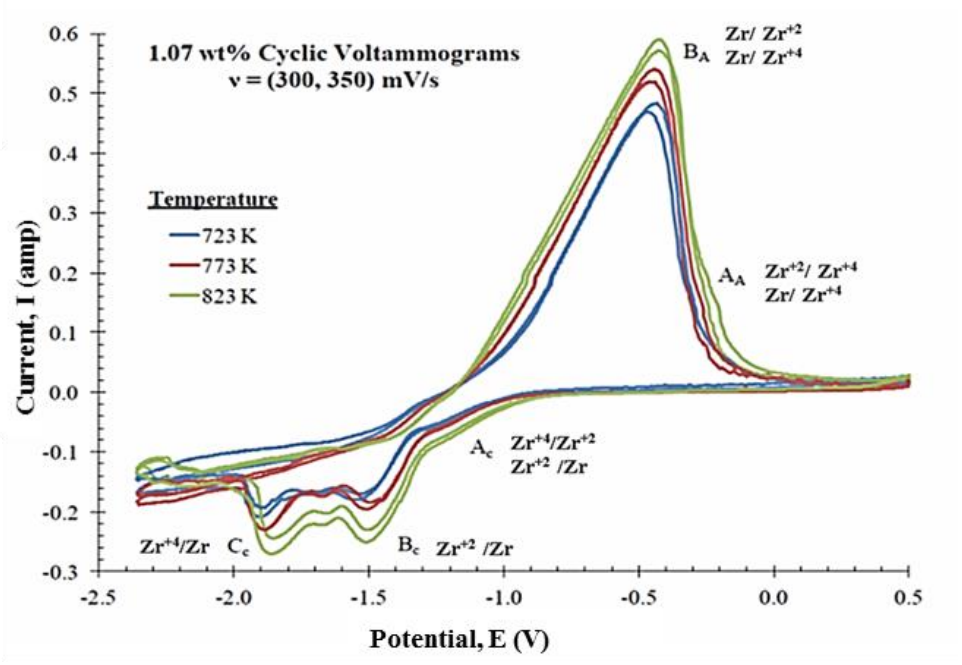

Fig. 3.9 Cyclic voltammetry of $ZrCl_4$ at different temperatures for 300 mV/s and 350 mV/s.

When the scan is being reversed, the first oxidation reaction peak is related to the oxidation of Zr/Zr^{+2} or Zr/Zr^{+4} or combination both reactions. Then, the next oxidation occurs at a more positive potential yielding two possible competing reactions: Zr/Zr^{+4} and Zr^{+2}/Zr^{+4}. In this part of the work, diffusion model has been applied on the zirconium chloride to clarify the probabilities of reaction mechanisms for each peak suggested by the literatures and displayed in Fig. 3.9. For this purpose, the initial guesses for diffusion coefficients and formal potentials at cathodic and anodic peaks (B_a, B_c and C_c in Fig.3.9) were assumed based on the value reported in literatures; these values are summarized in Tables 3.5 and 3.6.

The procedure is to adjust the values of diffusion coefficients and formal potentials for each peak and to finalize closely match the experimental peak. For this purpose, within the algorithm of simulation which is similar to uranium procedure, the diffusion coefficient and formal potential for each peak were tuning by a step of 10^{-7} cm^2/s and 2×10^{-4} V, respectively. That is, if the difference between the simulated and experimental anodic peak for current is greater than 0.001, the anodic diffusion coefficient would be decreasing by 10^{-7}, which is shown in Fig. 3.10. For the cathodic side, the diffusion coefficient would be increasing by 10^{-7}. In addition, if the simulated potential value differed from the experimental value, the anodic formal potential would be set to decrease by 2×10^{-4} (this is also the same at cathodic side). The number of iteration was the only difference between zirconium and uranium algorithms. That is, for zirconium, the iteration could be reaching up to 1700 times in order to achieve accurate results with a maximum error of ~0.4% and ~7% for current and potential, respectively. In general, the highest iteration was due to an anodic peak which was somehow the most complicated peak to properly fit. However, diffusion coefficient and formal potential for uranium chloride would be adjusted by less than 10 iterations. Thus, the processing time to simulate cathodic peaks for zirconium was generally lasted about 10

minutes and for highest anodic peak, it could take up to 2 to 3 hours. The results of predicted probability reactions during the zirconium chloride process based on diffusion model will be discussed in Section 3.4 where the area ratio for predicted model and experimental data will also be discussed.

Table 3.5 Initial guess of D and E^0 for $\mathbf{B_c}$ and $\mathbf{B_a}$ peak of 1.07, 2.48, and 4.98 wt% $ZrCl_4$ at different scan rates at 773 K, ^{*a}: Our Previous Work, and ^{*b}: Trial and Error.

Wt%	Peak	Scan rate (mV/s)	Diffusion Coefficient, D (cm^2/s)	Formal potential, E^0 (V)
1.07	Zr^{+2}/Zr	150, 200, 250, 300, 350	6.07×10^{-6}	-1.395*[a]
	Zr/Zr^{+2}	150, 200, 250, 300, 350	2.73×10^{-4}	-0.48*[a]
2.49	Zr^{+2}/Zr	100, 150, 200, 300	6.07×10^{-6}	-1.395[*a]
	Zr/Zr^{+2}	100, 150, 200, 300	2.73×10^{-4}	-0.48[*a]
	Zr^{+2}/Zr	250	1.00×10^{-5}	-1.58[*b]
	Zr/Zr^{+2}	250	1.60×10^{-4}	-0.15[*b]
4.98	Zr^{+2}/Zr	100, 150, 200, 300	6.07×10^{-6}	-1.60[*a]
	Zr/Zr^{+2}	100, 150, 200, 300	2.73×10^{-4}	-0.48[*a]

Table 3.6 Initial guess of D and E^0 for $\mathbf{C_c}$ peak simulation of 1.07, 2.48, and 4.98 wt% $ZrCl_4$ at different scan rates at 773 K.

Wt%	Peak	Scan rate (mV/s)	Diffusion Coefficient, D (cm^2/s)	Formal potential, E^0 (V)
1.07	Zr^{+4}/Zr	150, 200, 250, 300, 350	1.38×10^{-5}	-1.80
	Zr/Zr^{+4}	150, 200, 250, 300, 350	2.80×10^{-5}	-0.51
2.49	Zr^{+4}/Zr	100, 150, 200, 300	2.40×10^{-5}	-2.00
	Zr/Zr^{+4}	100, 150, 200, 300	2.80×10^{-5}	-0.51
4.98	Zr^{+4}/Zr	100	2.50×10^{-5}	-2.21
	Zr/Zr^{+4}	100	2.80×10^{-5}	-0.51
	Zr^{+4}/Zr	150, 200	2.10×10^{-5}	-1.99
	Zr/Zr^{+4}	150, 200	2.80×10^{-5}	-0.51
	Zr^{+4}/Zr	300	2.23×10^{-5}	-2.10
	Zr/Zr^{+4}	300	2.80×10^{-5}	-0.51

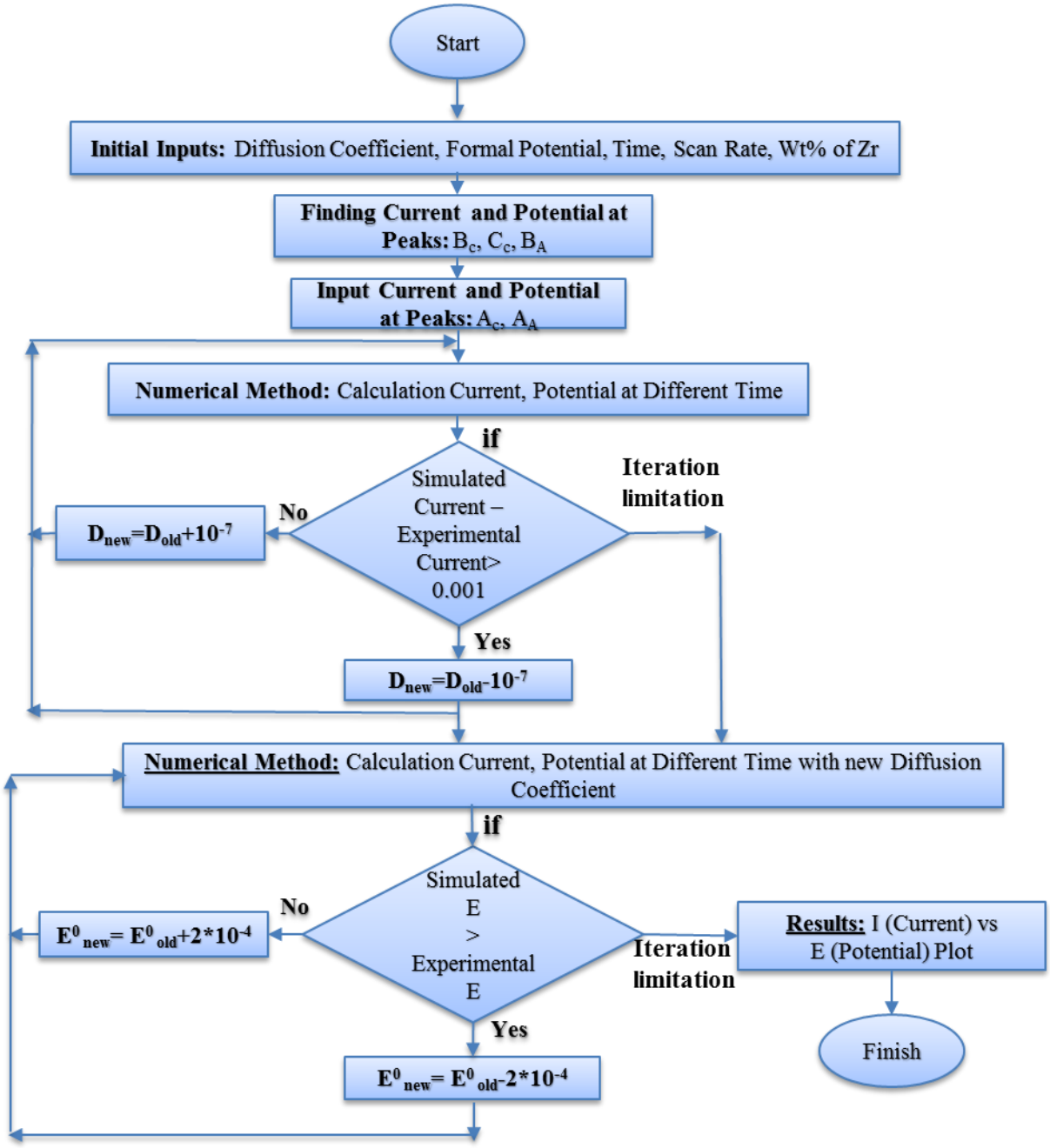

Fig. 3.10 Block diagram for anodic side.

3.4.2 Results

All reactions at each peak of zirconium chloride CV are illustrated in Fig. 3.9. By simulating the Zr^{+4}/Zr^{+2} and Zr^{+2}/Zr reactions separately at A_c peak based on the reported experimental data, it can be concluded that the combination of Zr^{+2}/Zr and Zr^{+4}/Zr^{+2} is occuring at this peak (see Fig. 3.11). First, the simulated data with literature diffusion coefficient of Zr^{+4}/Zr^{+2} reaction is not proper due to potential increased from -1.2 V instead of gradually decreasing (Fig. 3.11 (a)). Second, the diffusion coefficient which gives the best result (see Fig.3.11 (b)) is different from diffusion coefficient for peak B_c which is related to Zr^{+2}/Zr. Simulated results can be distinguished by blue color and are compared with experimental data sets showing in black color (Refs. 6, and 9). Fig. 3.12 further illustrates that the main reaction at A_a peak between Zr^{+2}/Zr^{+4} and Zr/Zr^{+4} is relied on the oxidation of Zr^{+2}/ Zr^{+4}. One important argument is related to anodic peak (B_a). Fig. 3.13 (a) shows a narrow plot coverage for 100% Zr/Zr^{+4}. If the anodic peak is related to 100% Zr/Zr^{+2}, the plot cannot cover the area under the anodic peak (see Fig. 3.13 (b)). Therefore, the combination of Zr/Zr^{+4} and Zr/Zr^{+2} was being considered and accomplished by determining a maximum ratio of area under anodic peak for experimental to simulated data. Figs. 3.14 displays the maximum coverage is discovered with 70% Zr/Zr^{+4} and 30% Zr/Zr^{+2} for both 1.07 wt% and 2.49 wt% $ZrCl_4$. Tables 3.7 and 3.8 provide the error values for the area ratio with 70% Zr/Zr^{+4} and 30% Zr/Zr^{+2} for both 1.07 wt% and 2.49 wt% $ZrCl_4$ will deliver optimum calculated current and potential values. Details for other scan rates are reported in Appendix V.

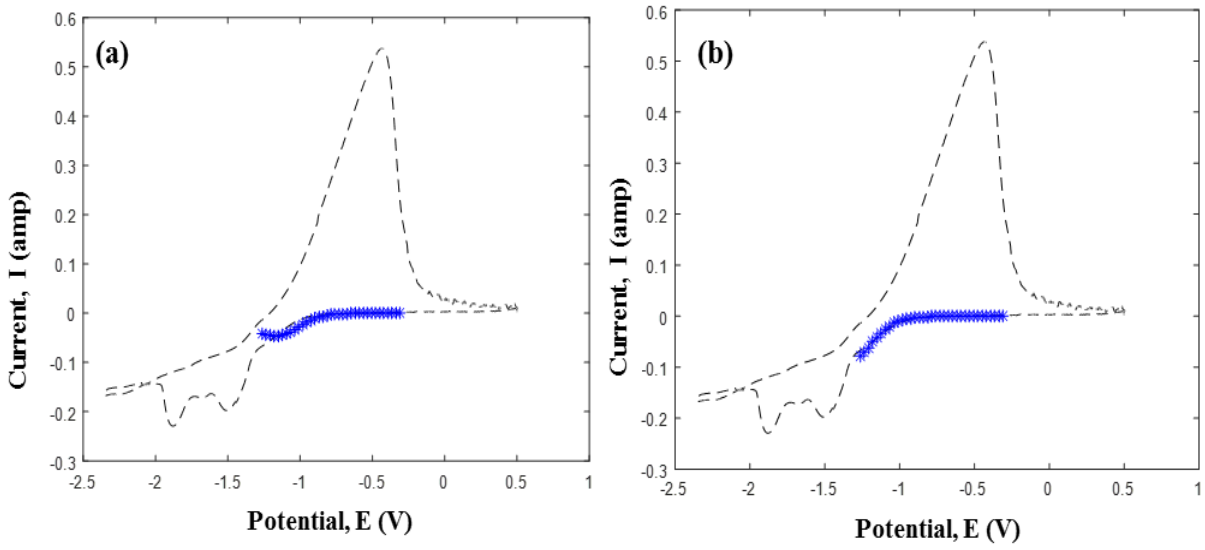

Fig. 3.11 Reaction probability at $\mathbf{A_c}$ peak for 1.07 wt% $ZrCl_4$ with 350 mV/s at 773 K, (a) Zr^{+4}/Zr^{+2} reaction, and (b) combination of Zr^{+4}/Zr^{+2} and Zr^{+2}/Zr.

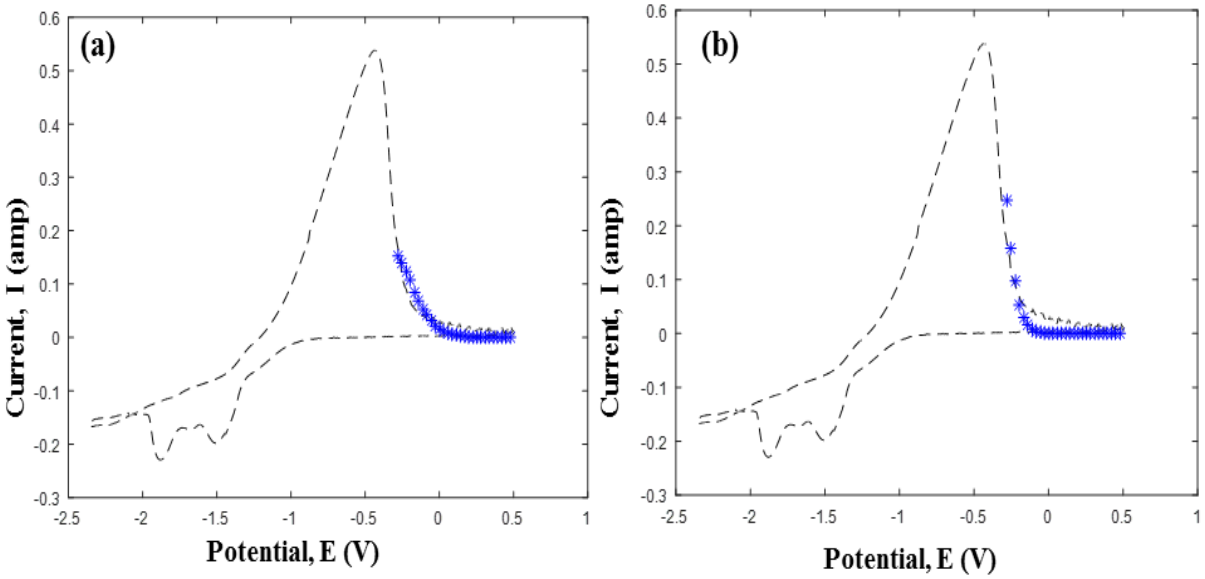

Fig. 3.12 Reaction probability at $\mathbf{A_a}$ peak for 1.07 wt% $ZrCl_4$ with 350 mV/s at 773 K, (a) Zr^{+2}/Zr^{+4}, (b) Zr/Zr^{+4}.

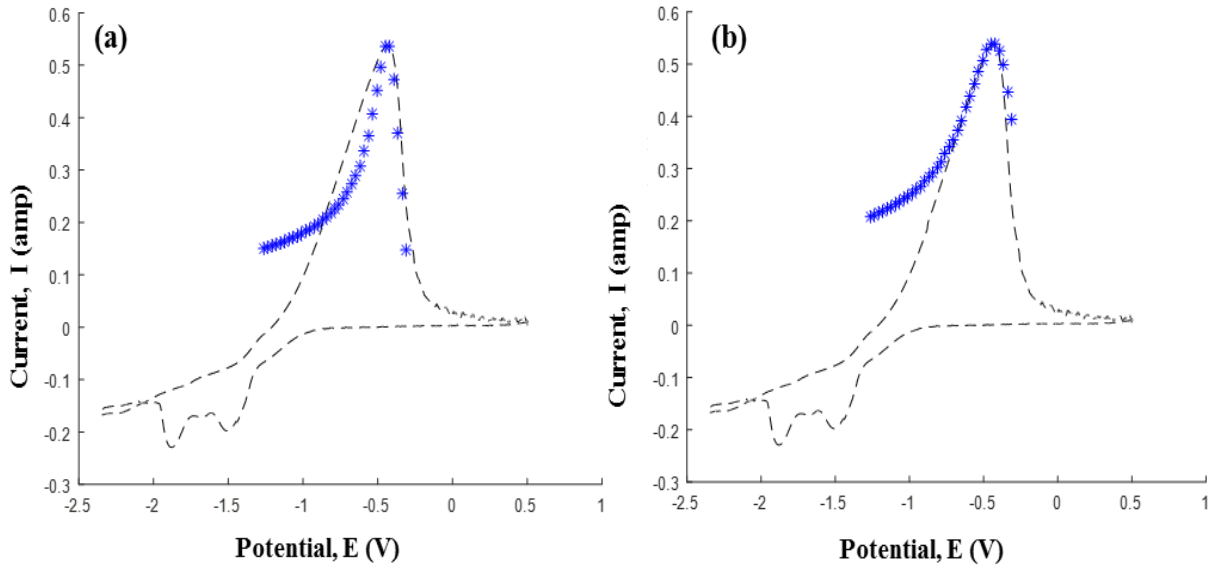

Fig. 3.13 Reaction probability at $\mathbf{B_a}$ peak for 1.07 wt% $ZrCl_4$ with 350 mV/s at 773 K, (a) 100% Zr/Zr^{+4}, (b) 100% Zr/Zr^{+2}.

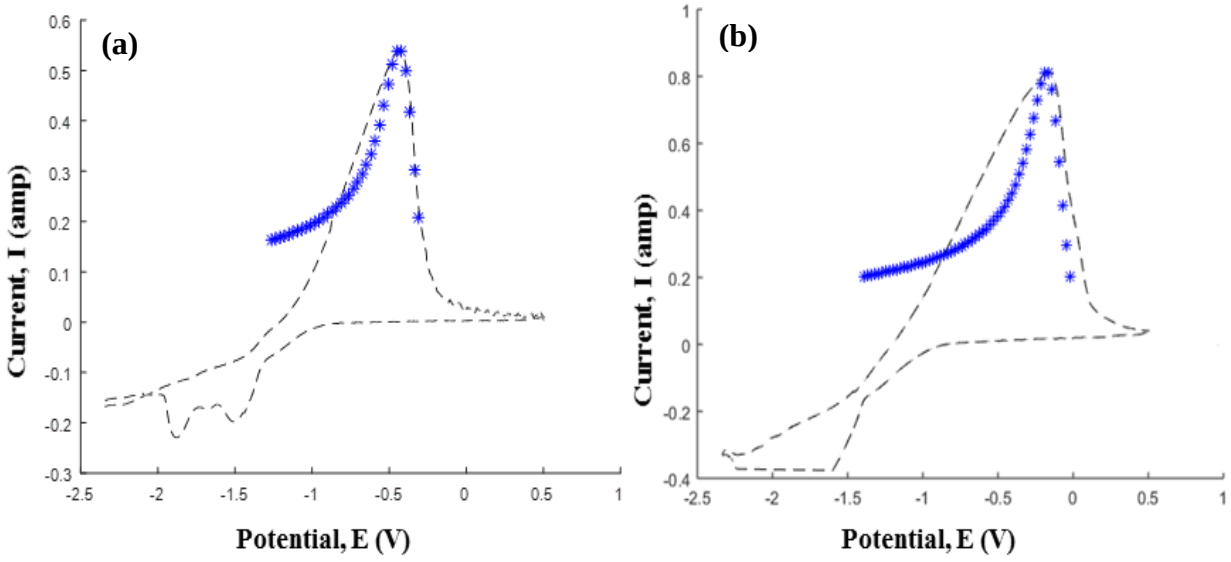

Fig. 3.14 Combination of 70% Zr/Zr^{+4} and 30% Zr/Zr^{+2} at $\mathbf{B_a}$ peak at 773 K, (a) 1.07 wt% $ZrCl_4$ with 350 mV/s, (b) 2.49 w% $ZrCl_4$ with 300 mV/s.

Table 3.7 Area ratio, current and potential error at $\mathbf{B_a}$ peak for 1.07 wt% $ZrCl_4$ with 300 mV/s and 350 mV/s at 773 K.

Zr^{+4}	Zr^{+2}	Area Ratio	Current Error (%)	Potential Error (%)	Area Ratio	Current Error (%)	Potential Error (%)
		1.07 wt%- 300 mV/s			**1.07 wt%- 350 mV/s**		
100	0	0.9268	0.5604	4.6974	0.9197	0.3125	3.6237
90	10	0.9013	0.4354	4.6974	0.8999	0.3014	2.8225
80	20	0.8775	0.0693	0.3663	0.8801	0.1808	3.6237
70	**30**	**0.8497**	**0.0517**	**0.3663**	**0.8574**	**0.0308**	**3.6237**
60	40	0.8265	0.1583	0.3663	0.8361	0.0900	3.6237
50	50	0.8025	0.2027	4.6974	0.8174	0.0079	2.8225
40	60	0.7772	0.1099	4.6974	0.7965	0.0282	2.8225
30	70	0.7538	0.0751	4.6974	0.7758	0.1834	3.6237
20	80	0.7281	0.0076	0.3663	0.7531	0.0138	3.6237
10	90	0.7037	0.0099	4.6974	0.7317	0.0630	3.6237
0	100	0.6776	0.0417	4.6974	0.7099	0.0272	3.6237

Table 3.8 Area ratio, current and potential error at $\mathbf{B_a}$ peak for 2.49 wt% $ZrCl_4$ with 100 and 300 mV/s at 773 K.

Zr^{+4}	Zr^{+2}	Area Ratio	Current Error (%)	Potential Error (%)	Area Ratio	Current Error (%)	Potential Error (%)
		2.49 wt%- 100 mV/s			**2.49 wt%- 300 mV/s**		
100	0	1.1824	0.0424	0.3304	1.1177	0.3889	23.2214
90	10	1.1523	0.1445	0.3390	1.0775	0.0431	0.9890
80	20	1.1233	0.2420	0.3478	1.0569	0.4402	0.9890
70	**30**	**1.0927**	**0.2142**	**0.3575**	**1.0335**	**0.3912**	**0.9890**
60	40	1.0626	0.2063	0.3676	1.0053	0.0119	15.7729
50	50	1.0334	0.1368	0.3780	0.9769	0.1091	0.9890
40	60	1.0029	0.0729	0.3895	0.9489	0.1363	0.9890
30	70	0.9707	0.1551	0.4024	0.9226	0.1740	15.7729
20	80	0.9396	0.1209	0.4158	0.8958	0.0806	15.7729
10	90	0.9075	0.1349	0.4304	0.8669	0.0717	0.9890
0	100	0.8762	0.1332	0.4459	0.8394	0.0827	0.9890

The current and potential errors, and area ratios for combination of 70% Zr/Zr^{+4} and 30% Zr/Zr^{+2} at B_a peak for 1.07 wt% and 2.49% at different scan rates are listed in Tables 3.9 and 3.10, respectively. The combination of highest anodic peak for 4.98 wt% is different from other concentrations. Although by considering 70% Zr/Zr^{+4} and 30% Zr/Zr^{+2}, which gives the ratio area closer to one (see Fig. 3.15(a)), this combination still does not cover the area properly. Therefore, another combination of 30% Zr/Zr^{+4} and 70% Zr/Zr^{+2} was being considered. The result of using that combination is shown in Fig. 3.15(b). Here, the covered area is improved with the current error of ~0.2% and a negligible change in the potential. The potential and current error, and area ratio for combination of 70% Zr/Zr^{+4} and 30% Zr/Zr^{+2} and 30% Zr/Zr^{+4} and 70% Zr/Zr^{+2} for 4.98 wt% at 200 mV/s are reported in Table 3.11. Details related to other scan rates are listed in Appendix V.

Table 3.9 Area ratio, current and potential error for combination of 70% Zr/Zr^{+4} and 30% Zr/Zr^{+2} at $\mathbf{B_a}$ peak for 1.07 wt% $ZrCl_4$ at 773 K with different scan rates.

Scan rate (mV/s)	Area Ratio	Current Calculation (amp)	Current Raw-data (amp)	Current Error (%) (V)	Potential Calculation (V)	Potential Raw-data (V)	Potential Error (%) (V)
250	0.6597	0.4994	0.4989	0.1124	-0.4888	-0.4889	0.0189
300	0.8497	0.5269	0.5266	0.0517	-0.4517	-0.4534	0.3663
350	0.8574	0.5384	0.5382	0.0308	-0.4221	-0.4374	3.6237

Table 3.10 Area ratio, current and potential error for combination of 70% Zr/Zr^{+4} and 30% Zr/Zr^{+2} at B_a peak for 2.49 wt% $ZrCl_4$ at 773 K with different scan rates.

Scan rate (mV/s)	Area Ratio	Current Calculation (amp)	Current Raw-data (amp)	Current Error (%) (amp)	Potential Calculation (V)	Potential Raw-data (V)	Potential Error (%) (V)
100	1.0927	0.6390	0.6376	0.2142	-0.3447	-0.3442	0.3575
150	1.0383	0.7114	0.7116	0.0219	-0.2620	-0.2690	2.6712
200	1.0545	0.7511	0.7511	0.000657	-0.2093	-0.2224	6.2431
250	0.9853	0.7823	0.7825	0.0307	-0.1874	-0.1957	4.4475
300	1.0335	0.8120	0.8152	0.3912	-0.1658	-0.1641	0.9890

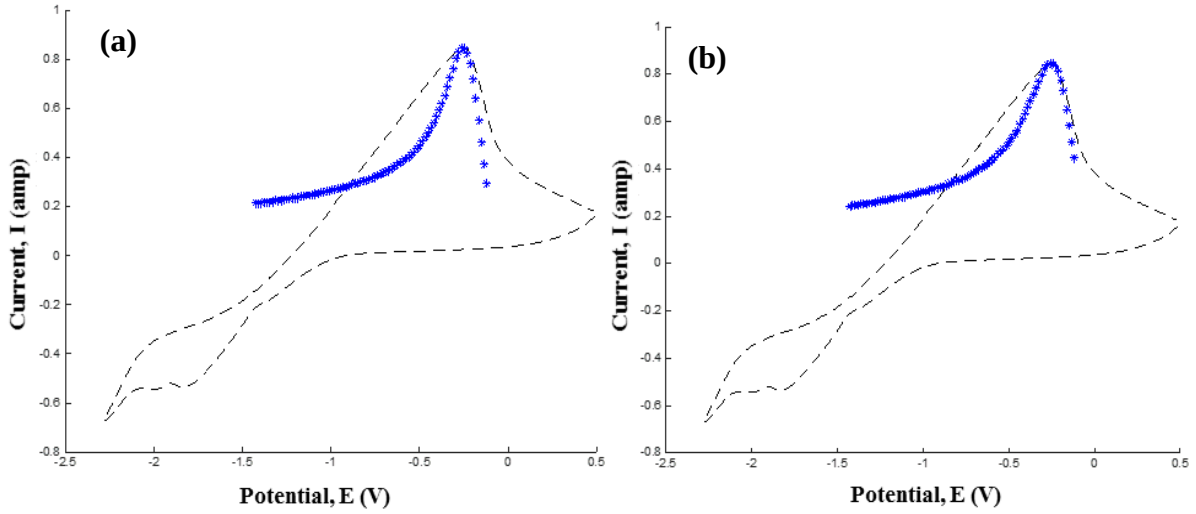

Fig. 3.15 4.98 wt% $ZrCl_4$ with 200 mV/s at 773 K, (a) combination of 70% Zr/Zr^{+4} and 30% Zr/Zr^{+2} at $\mathbf{B_a}$ peak, (b) combination of 30% Zr/Zr^{+4} and 70% Zr/Zr^{+2}.

Table 3.11 Comparison the current error, potential error and the area ratio for 70% Zr/Zr^{+4} and 30% Zr/Zr^{+2} (Denoted by X) versus 30% Zr/Zr^{+4} and 70% Zr/Zr^{+2} (Denoted by Y) of 4.98 wt% $ZrCl_4$ at 200 mV/s and 773 K.

Reaction Type	Area Ratio	Current Calculation (amp)	Current Raw-data (amp)	Current Error (%) (amp)	Potential Calculation (V)	Potential Raw-data (V)	Potential Error (%) (V)
X	0.997	0.8444	0.8478	0.4060	-0.2433	-0.2497	2.6354
Y	0.892	0.8471	0.8478	0.0858	-0.2593	-0.2497	3.6980

3.5 Conclusion

In summary, the reverse-engineering of CV model demonstrated an ability of predicting and tracking of the current versus potential graph and concentration versus time graph. The outcomes show that the model can regionally trace the CV with a low RMS error. However, there is a limitation at the irreversible anodic side which does not have any real impact on the conclusion. This limitation can perhaps be related to the graphical shape which is sharp and is not Gaussian in nature. The code run time is approximately two minutes with an adequate time interval of 0.08 seconds providing a proper robustness for a near real-time detection technique. The concentration of each species at the reversible and irreversible parts of the anodic and cathodic sides can be calculated and are illustrated based on increasing time which provided a visual representation of the whole process. These results also can be performed in the GUI environment. In addition, by selecting the current versus potential graph in *Matlab*, the concentration of reductant and oxidant species at each point can be computed.

Also, we have shown the capability of predicting the probability reactions during the zirconium chloride process using the the reverse-engineering CV method. The main reactions

occurring at cathodic side were the combination of Zr^{+2}/Zr and Zr^{+4}/ Zr^{+2} for the first peak, Zr^{+2}/Zr for the second peak, and Zr^{+4}/Zr for the third peak. At the anodic section, the first oxidation reaction was related to the combination of Zr/Zr^{+4} and Zr/Zr^{+2}. This conclusion is based on the optimum area ratio under B_a peak and the minimum potential and current errors. For the 1.07 wt% and 2.49 wt% zirconium chloride at 773 K, the main reaction at highest anodic peak is the combination of 70% Zr/Zr^{+4} and 30% Zr/Zr^{+2} at different scan rates. However, the optimum reaction at this peak for 4.98 wt% is switched to the combination of 30% Zr/Zr^{+4} and 70% Zr/Zr^{+2} at different scan rates. For all concentrations, the difference between calculated current and the experimental values are ~0.4% with the potential error of ~7% at the most. This complexity has shown the limitation of the modified diffusion model reaches on its accuracy and predictability. Therefore, another method must be considered in order to provide a robust simulation and prediction of the CV data sets. Thus, an artificial neural intelligent (ANI) methodology has been proposed as a novel alternative method and will be explored next.

Chapter 4: Artificial Neural Intelligent (ANI) and Results of ANI[2]

4.1 Background and Theories

Artificial Neural Intelligent (ANI) is a novel data analysis and simulation method that can be applied to electrochemical data sets and is inspired by brain neural neurons (Refs. 49, and 50). Due to the similarity between a computer machine and biological nervous system, it has been discovered that a computer has the capability of learning by training samples (Ref. 50). ANI could be implemented to learn massive training data set through iterations and interrelationships among system variables such as currents, potentials, concentrations, scan rates, processing times, and weight percent, without requiring the specific knowledge to predict the desire cyclic voltammetry (CV) graph which were not explicitly trained (Refs. 49, 51, and 52). One uniqueness of using ANI is its capability with non-linear, noisy, and uncertain data sets which is invaluable for modeling, prediction, and optimization towards detection and material accountability in nuclear safeguards (Refs. 49 to 53).

ANI is consisted of one input layer, hidden layers, and one output layer, which are interconnected by a number of nodes called neurons. One simplest type of ANI that information goes in one direction with no loop or cycle is called feedforward. And one of the simplest type of

[2] Content in Chapter 4 are cited from the author's publication:
S. Rakhshan Pouri, M. Manic, and S. Phongikaroon," A Novel Framework for Intelligent Signal Detection via Artificial Neural Networks for Cyclic Voltammetry in Pyroprocessing Technology", Submitted to *Annals of Nuclear Energy.*

feedforward network which has been used widely is perceptron (See Fig. 4.1). The inputs (x_1, x_2, x_j) forward to one node and provide a single output (Ref. 54). The weighted inputs are added and compared with a threshold value; then it will return an output as 0 or 1. If the weighted sum is less than a given threshold ($\sum_j w_j x_j \leq threshold$), the output returns 0 and if it is greater than the threshold ($\sum w_j x_j > threshold$), the output will return 1 (Ref. 54).

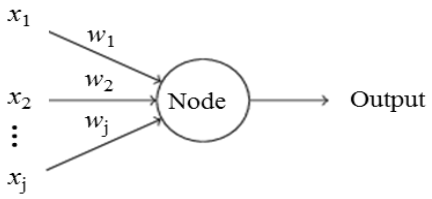

Fig. 4.1 Perceptron schematic (Ref. 54).

To simplify the threshold condition in perceptron, the bias value (b) is used. This value can be thought of as how easy is to get 1 value at the output (Ref. 54), which can be describe as:

$$\text{output} = \begin{cases} 0 & \text{if } \sum_j w_j x_j + b \leq 0 \\ 1 & \text{if } \sum_j w_j x_j + b > 0 \end{cases} \tag{4.1}$$

In reality, the system is a complex network of perceptrons that are are being required to make a suitale decision. A multi-layer perceptron (MLP) is illustrated in Fig. 4.2, which is consisted of different hidden layers (Ref. 51).

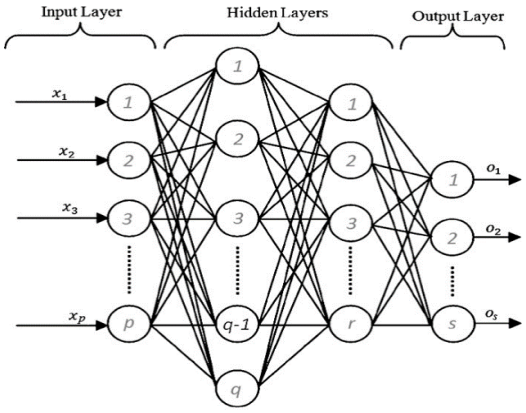

Fig 4.2 Multi- layer perceptron schematic (Ref. 51).

Inputs data using the MLP network are weighted (w_{ji}) and summed up with the constant bias term (b_i), as shown in Fig. 4.3 (Ref. 55). This approach yields the resulting data (n_i) input to the activation function ($g(n_i)$), giving the outputs (y_i) (Ref. 55). The hidden layer comprises of neurons arrays that are received, transformed, and transferred the signal from the previous layer. The signals from the input and hidden layer to the output layer were modeled by an activation function which is generally linear, hyperbolic tangent, and sigmoid (Ref. 56). Due to the fact that the most productive activate function for the MLP is related to sigmoid function, the ANI feature in the *Matlab* software is written based on the sigmoid function.

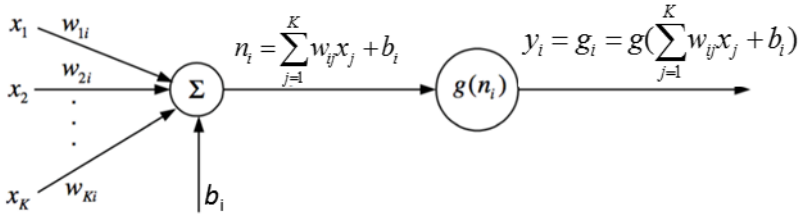

Fig 4.3 A multilayer perceptron network with one hidden layer (Ref. 55).

Althought the ANI is improved by using the bias value instead of threshold, a small change in the weight or bias causes the output flip from 1 to 0 and vice versa. For the MLP network, this

issue can be solved by using the sigmoid neurons that gives the output 0, 1 and the value from 0 to 1 (Ref. 54); here, Fig. 4.4 displays the sigmoid function. Mathematically, the sigmoid function can be defined as:

$$g(n_i) = \frac{1}{1+e^{-n_i}} = \frac{1}{1+\exp(\sum_{j=1}^{K} w_{ij}x_j + b_i)} \tag{4.2}$$

Here, the experimental data sets (input data) were divided into three main parts: (1) training data set which was a partial of whole experimental data sets for adjusting the weights and bias; (2) validation data set which is an independent data set from training sample but can be regrated as a part of traning data sets because it has been used in training phase to minimize the overtratinig; and (3) the leftover data sets were related to the test data sets to assess the system performance.

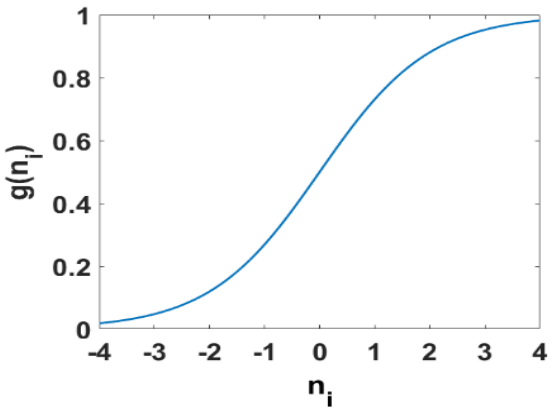

Fig. 4.4 Sigmoid function schematic.

In theory, overfitting happens when the system begins to memorize the training data set rather than learning (Ref. 57); that is, the validation error starts to increase after an optimal situation (see Fig. 4.5) and the training error goes down gradually while the test error increases progressively (Refs. 58, and 59). Fig. 4.5 shows that the best predictive model is where the validation error (ε) reaches a global minimum (Ref. 60). Adding of additional hidden layer and increasing the number

of neurons within each layer enhanced the neural network complexity are expected to improve prediction resulting in a lower error for a fixed training data set. It is important to consider that if the number of layers goes up to four layers, the overfitting can occur and the run time increases significantly, therefore defeating the purpose of achieving a fast and robust detection method. For this reason, validation checks have been considered which represent the numbers of consecutive iterations that system performance fails to decrease. The use of validation here is related to ANI assessment and must not be confused by verification and validation (V&V) (Ref. 61).

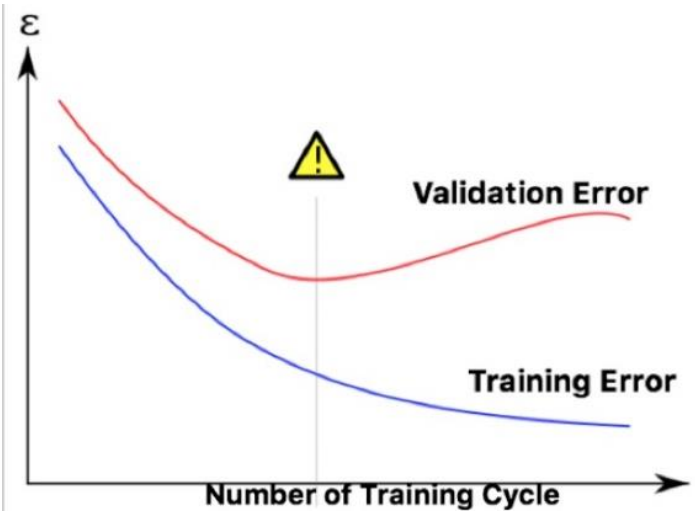

Fig 4.5 Overfitting in learning (Ref. 60).

Over the recent decades, various algorithms for determining the network parameters such as weight values have been developed. Based on the literature reported, the most well-known is back-propagation algorithm (BPA) and Levenberg-Marquardt algorithm (LMA). Here, LMA is more efficient due to its fast process time and can provide an adequate way for curve-fitting problems because of interpolating between two method of Gauss-Newton algorithm (GNA) and Gradient Descent (Ref. 55 and 62). The gradient descent method can be used to find a local minimum of a function by reducing the sum of the squared errors with updating the parameters in

the steepest-descent direction. However, the sum of the squared errors in the Gauss-Newton method is reduced by assuming that the least squares function is locally quadratic—finding the minimum of the quadratic (Ref. 55).

4.2 Zirconium Chloride

4.2.1 Computational Procedure

ANI was implemented on the cyclic voltammetry (CV) to find a condition that provided a minimum error while predicting unseen data sets. A huge experimental data set of 0.5 to 5 wt% of $ZrCl_4$ in LiCl-KCl eutectic molten salt at 773 K under different scan rates (over 230,000 data points) collected by Hoover (Ref. 6) was being considered through the commercial software package, *Matlab*. Each experimental data set was consisted of the following variables—potential, and process time for different or concentrations and scan rates as the input data and current as the output. The input and output variables are shown in Fig. 4.6.

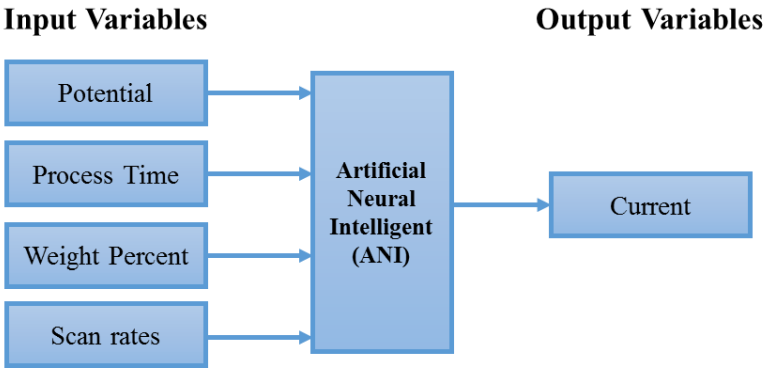

Fig. 4.6 Input and output variables of the ANI.

The overall goal was to determine the structures that ANI could be used to predict different systematic situations; these were (1) the minimum training data set requirement for achieving the

lowest error, (2) the adequate numbers of hidden layers, (3) neurons at each layer, and (4) number of validation checks with a minimum error. It was expected to apply this work to trace the operating current versus potential of a case with inadequate input information by interpolating between known information giving a low present error. Because this work focused on the minimum number of training data that could provide a reasonable predicted error, various training data set combinations were explored. This work could be implemented with 43% of total experimental data set at a specific and fix combination. These designs are listed in Table 4.1. Here, some conditions were being repeated two to three times (see Table 4.1). The training data sets are indicated in shade and the test data sets are indicated in clear-white. Two conditions of 0.5 wt% at 200 and 450 mV/s were considered as train and test samples for further discussion.

Table 4.1 Experimental data set for $ZrCl_4$ in LiCl-KCl at 773K.

Concen. (mol/cm^3)	**Scan Rate (mV/s) Condition**										
0.5wt%	200 Train	250 Test	300 Train	350 Test	400 Train	450 Test	500 Train				
1wt%	150 Train	150 Test	200 Train	200 Test	250 Train	250 Test	300 Test	300 Test	350 Train	350 Test	
2.5wt%	100 Train	100 Test	150 Test	200 Test	250 Train	300 Train	300 Test	400 Test	500 Train		
5wt%	50 Train	50 Test	100 Train	100 Test	150 Train	150 Test	200 Train	200 Test	200 Test	250 Test	250 Test
	300 Train	300 Test									

It is challenging to find out a suitable fixed combination data sets. However, if the train data sets contain different concentrations and scan rates, ANI can predict the output variables extremely well. For this purpose, the training data sets in this study have been selected in the way that they contain at the most 50% of each concentration condition. For example, in this study, the number of data sets for 0.5 wt% concentration was as low such that 57 percents of the experimental

data sets at this condition were selected as training data sets. In contrast, for 5 wt% with more data sets, only 38 percents of experimental data set were used. The selected scan rates conditions should be distributed to cover the slow and fast detections. It is a good idea to decide a scan rate step based on the experimental data sets. For example, in Table 4.1, the scan rate step was considered at 100 mV/s. However, 50, 100 mV/s, or combination of both could be applied on other concentrations. There is no specific rule to select the desired fixed condition. However, selecting the fixed condition with a simple stusture (eg. [7] with 10 validation checks) and training the data sets should illustrate which condition was not predicted accurately. Then, the combination of train and test should be modified.

As noted above, the input data were divided into two parts. After defining the number of training data sets, hidden layers, neurons at each layers, epochs, and validation checks, the data were scaled between -1 to 1 to improve the network training speed. It needs to mention that the data can not been scaled between 0 to 1 because the CV includes both positive and negative current values. After scaling, training data set was undergoing through a training process. At each epoch, the validation data set would control the overfitting. The train process would stop if it reached the minimum mean square error (MSE) between the simulated output and real data, or the defined epoch, or the validation checks. Fig. 4.7 shows the the flow diagram of ANI in this study; and the full code can be found in Appendix VI. The framework proposed in this paper entailed running ANI on different hidden layers (1 to 3) with various neurons (1 to 30) at several validation checks (1 to 30). The ANI routine was applied on one hidden layer with different neurons and each at different validation checks. The mean absolute percentage error (MAPE) between experimental and predicted data sets for 0.5 wt% at 200 and 450 mV/s was calculated using the following expression:

$$MAPE = \frac{100}{n}\sum_{t=1}^{n}\left|\frac{ActualValue_t - ForecastValue_t}{ActualValue_t}\right| \quad (4.3)$$

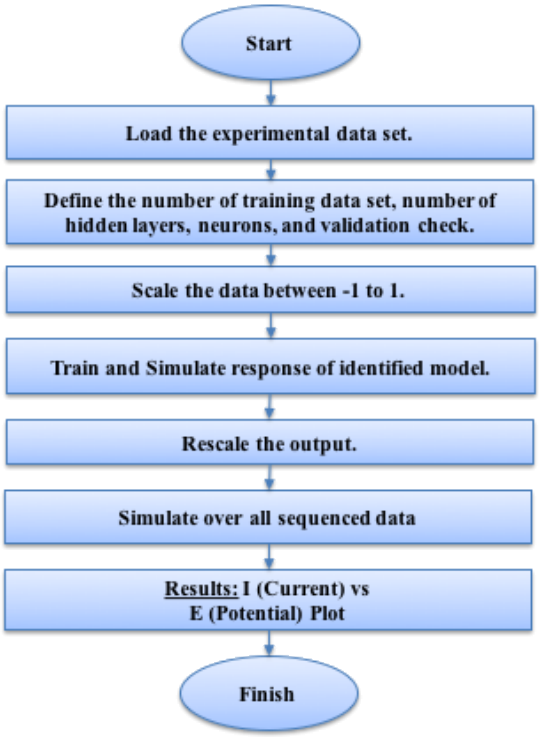

Fig.4.7 The flow diagram of ANI in this work.

Then, the structure that both cases (200 and 450 mV/s) provided a minimum average percent error was selected. Next, the number of hidden layer was increased to 2 and 3 layers following the same procedure. Thus, the situation that gave almost the same minimum average percent errors for both 200 and 450 mV/s was chosen and the predicted CV plots were compared and validated with the existing experimental data sets. The schematic flow diagram of the computational procedures is shown in Fig. 4.8.

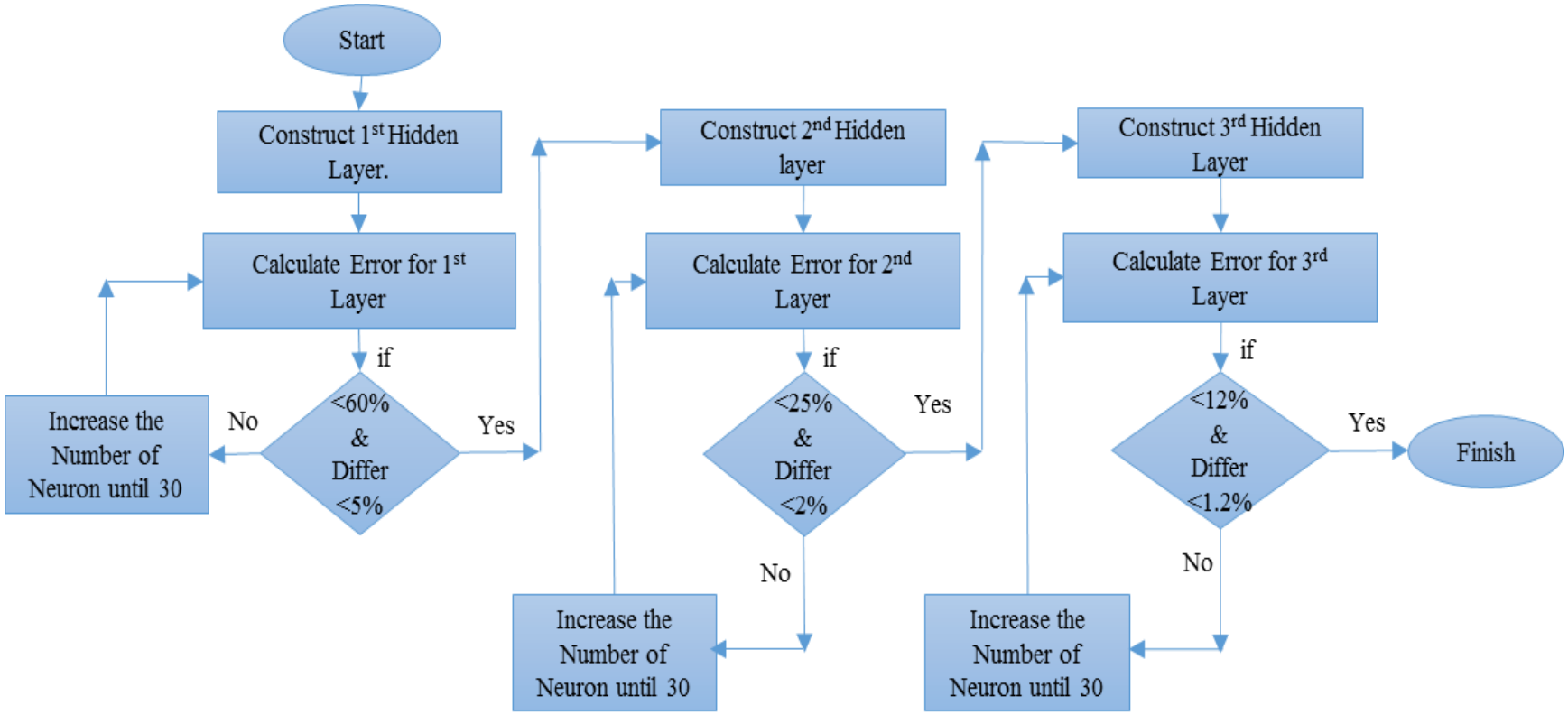

Fig. 4.8 Procedure flow chart.

In addition to the *Matlab* code, this work was also written in a GUI environment (see Appendix VI for details). The GUI layout for $ZrCl_4$ is indicated in Fig. 4.9. One of the GUI outcomes was CV graphs illustrating the predicted data through ANI in comparison to the real experimental data sets. The CV results will be discussed in Section 4.2.2.2.

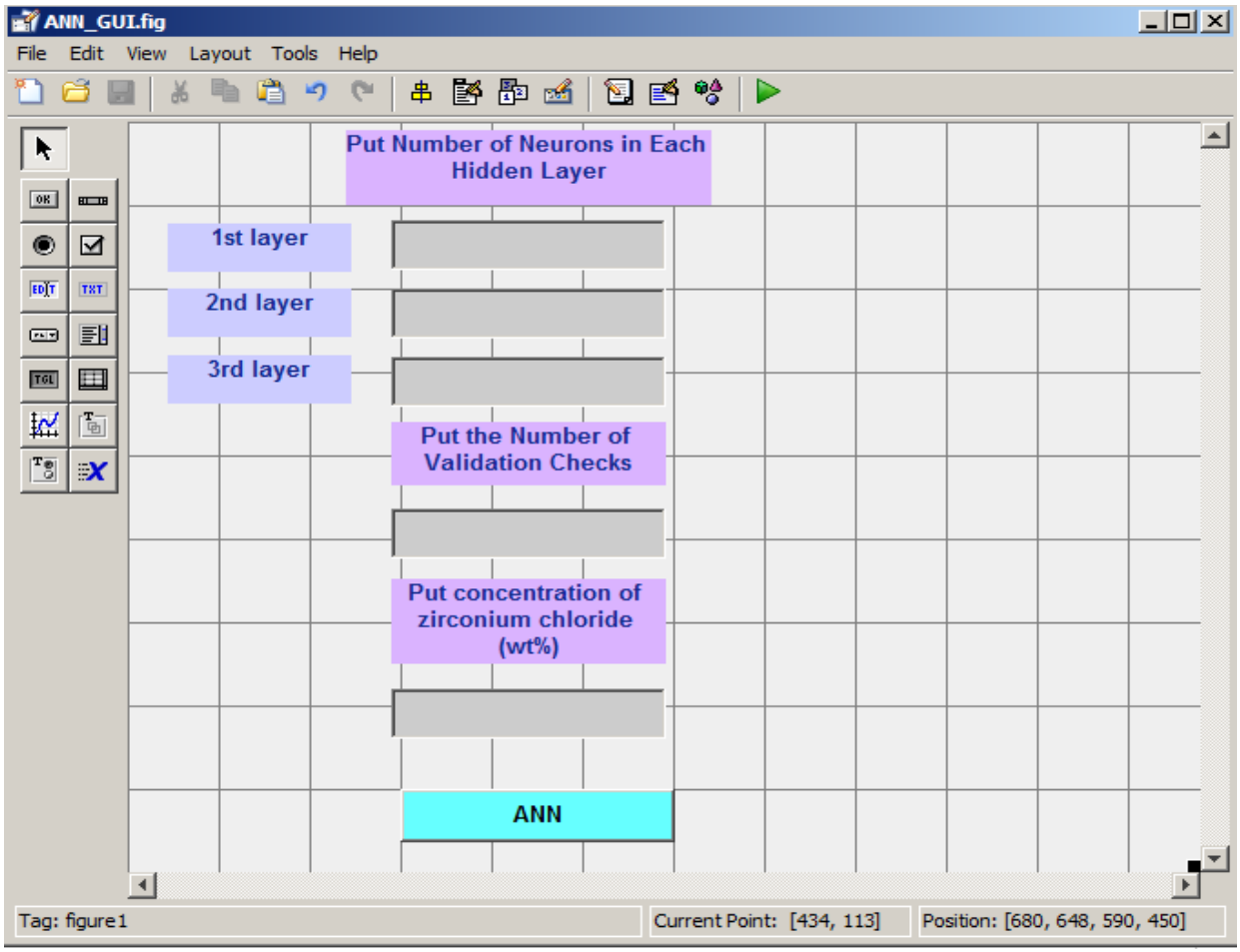

Fig. 4.9 GUI layput for ANI implementation of $ZrCl_4$ in LiCl-KCl eutectic.

By an example, the Matlab training platform can be clarified. Imagine there is a three hidden layers with 10, 11, and 25 neurons at each layer. The maximum number of epoch and the validation checks were defined as 5000, and 19, respectively. The training would stop after 13 iterations. Figures 4.10 and 4.11 display the neural network training and performance platform. Fig. 4.11 shows that the MSE deacreses gradually as the number of epochs increases.

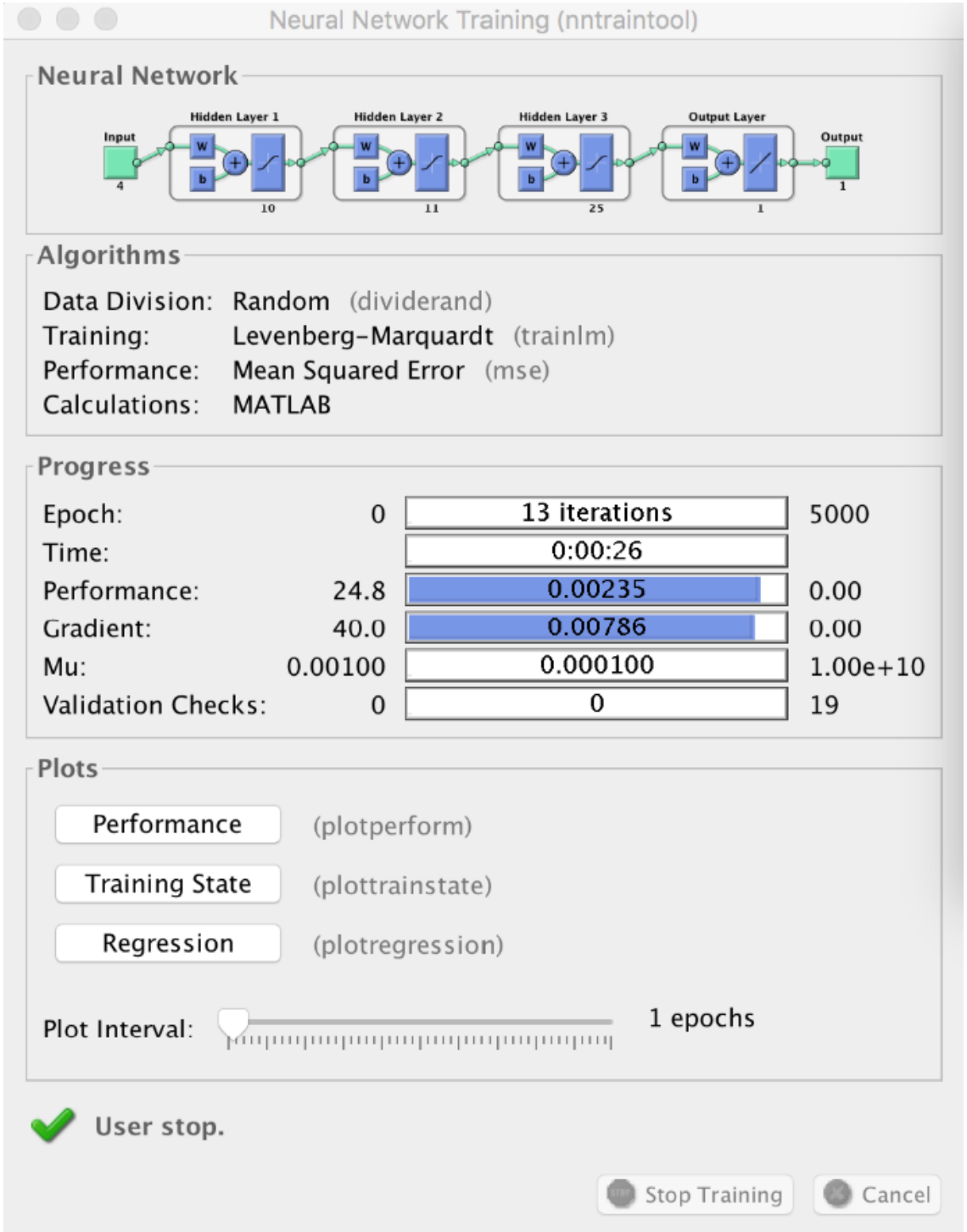

Fig. 4.10 The neural network training platform.

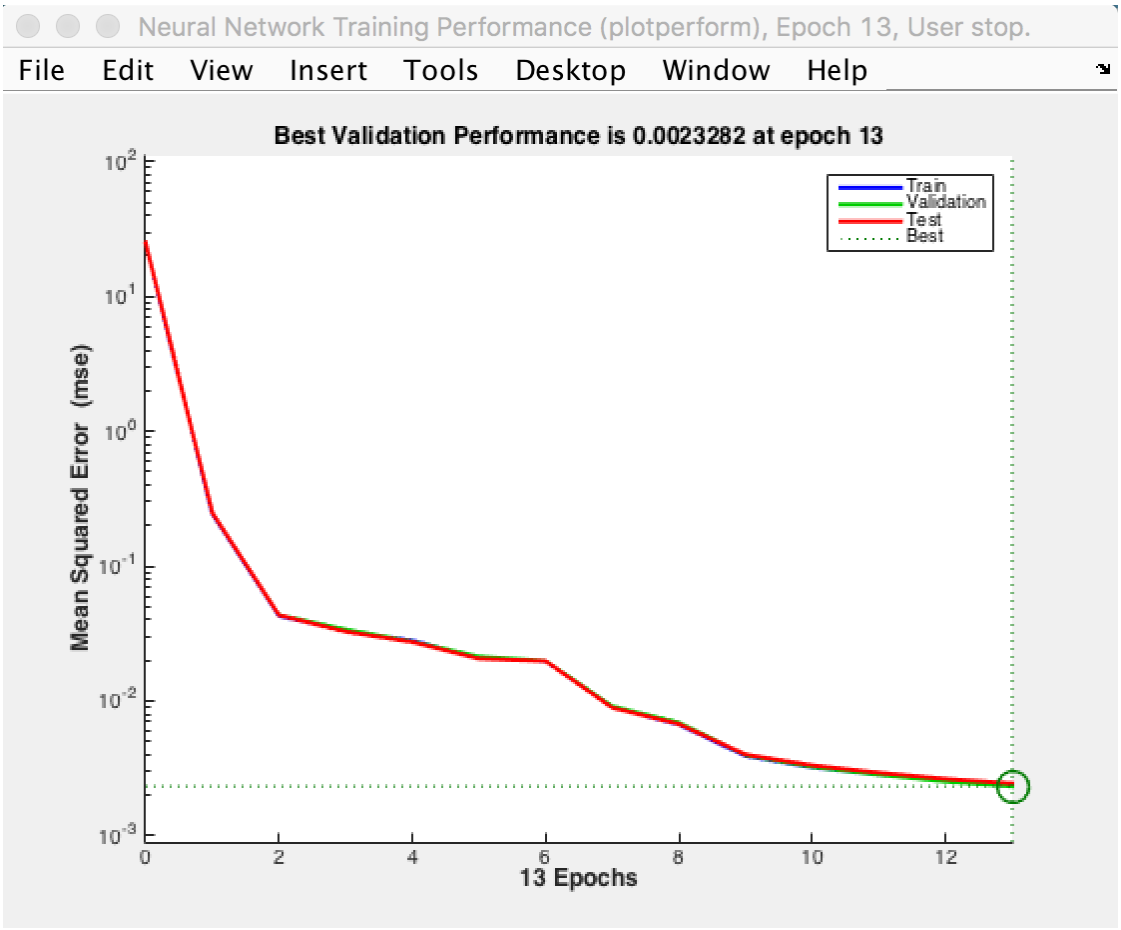

Fig. 4.11 The neural network performance platform.

4.2.2 Results

4.2.2.1 Determination of the First, Second and Third Hidden Layers

Fig. 4.12 shows the comparisons between minimum average percent errors of one hidden layer with various number of neurons and validation checks for 200 and 450 mV/s. Here, the minimum average error for one neuron at the first hidden layer with 1 to 30 validation checks for 200 and 450 mV/s are 96%, and 222%, respectively. The errors decrease to 45%, and 31% for 30

neurons at 200 mV/s and 450 mV/s, respectively. The points that both considered train and test samples provide the average error less than 60% while the deviations are less than 5% have been marked by the dashed circles. It can be seen that 8, 9, 10, and 25 neurons at the first layer (indicated by [8], [9], [10], and [25]) meet the mentioned criteria. It is important to mention that enhancing the number of neurons will also increase the processing time. For example, processing time of the first layer with 30 validation checks for 1 and 30 neurons are about 18 seconds and 8 minutes, respectively. The long processing time is the reason why the 25 neurons, which is marked in red (see Fig. 4.12), is not being considered for the second layer study.

To investigate the second hidden layer, the results from first hidden layer were selected as the starting point. First, we considered the case of having 8 neurons at the first hidden layer and 1 to 30 neurons at the second layer, denoted by [8, 1-30], with 1 to 30 validation checks. The points that provide average percent errors less than 25% for both 200 and 450 mV/s while having the difference around 2% are marked in Fig. 4.13. The results indicate that [8, 13], [8, 17], and [8, 30] fall within the criteria. Here, the processing time for [8, 1] at 30 validation checks is approximately 9 minutes and increases up to ~31 minutes for the [8, 30]. Therefore, the [8, 30] case was not selected for the third hidden layer study. As indicated in Figs. 4.13(a) – 4.13(c), [9, 13], [9, 15], [9, 21], [10, 6], [10, 11], and [10, 26] meet the mentioned criteria and can be considered for the next hidden layer.

All the selected results from the two layers were further studied for the third hidden layer. Criteria in this part were to select the points that both train and test samples would yield an average error below 12% with a difference of 1.2%; Fig. 4.14 displays the errors for [8, 13] and [8, 17], respectively. In addition, the [9, 13], [9, 15], [9, 21], [10, 6], [10, 11], and [10, 26] are shown in

Figs. 4.15 and 4.16. The red points indicate that these structures have not been considered as the final results due to their long processing times.

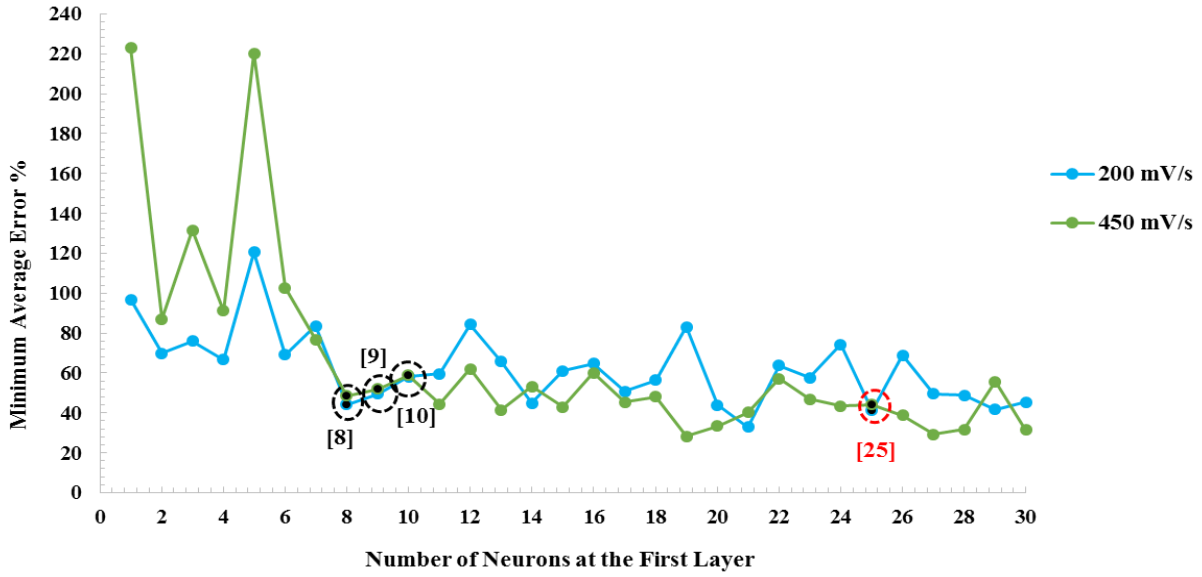

Fig. 4.12 One hidden layer with 1 to 30 neurons and 1 to 30 validation checks for 0.5 wt% at 200 mV/s and 450 mV/s (Black circle = short simulation time; Red circle = long simulation time).

Each point mentioned in Figs. 4.13 to 4.16 are related to a specific validation check. For example, the train and test sample points for [10, 26, 5] structure in Fig. 4.16 are occurred at 21 and 17 validation checks, respectively; this give the minimum average error of ~9%. Therefore, to select the proper validation check for [10, 26, 5] structure, we would routinely swab the validation checks to assure the average minimum error. That is, the train sample points for [10, 26, 5] structure would be verified by 17 validation checks and vice versa. The results of this reversal technique show the average error percent for train (using 17 validation checks) and test samples (using 21 validation checks) are 18% and 31%, respectively. Thus, by selecting the [10, 26, 5] with 17 validation checks, the average error for train and test samples are, 18% and 9%, respectively. These results yield a lower error in comparison to 21 validation checks.

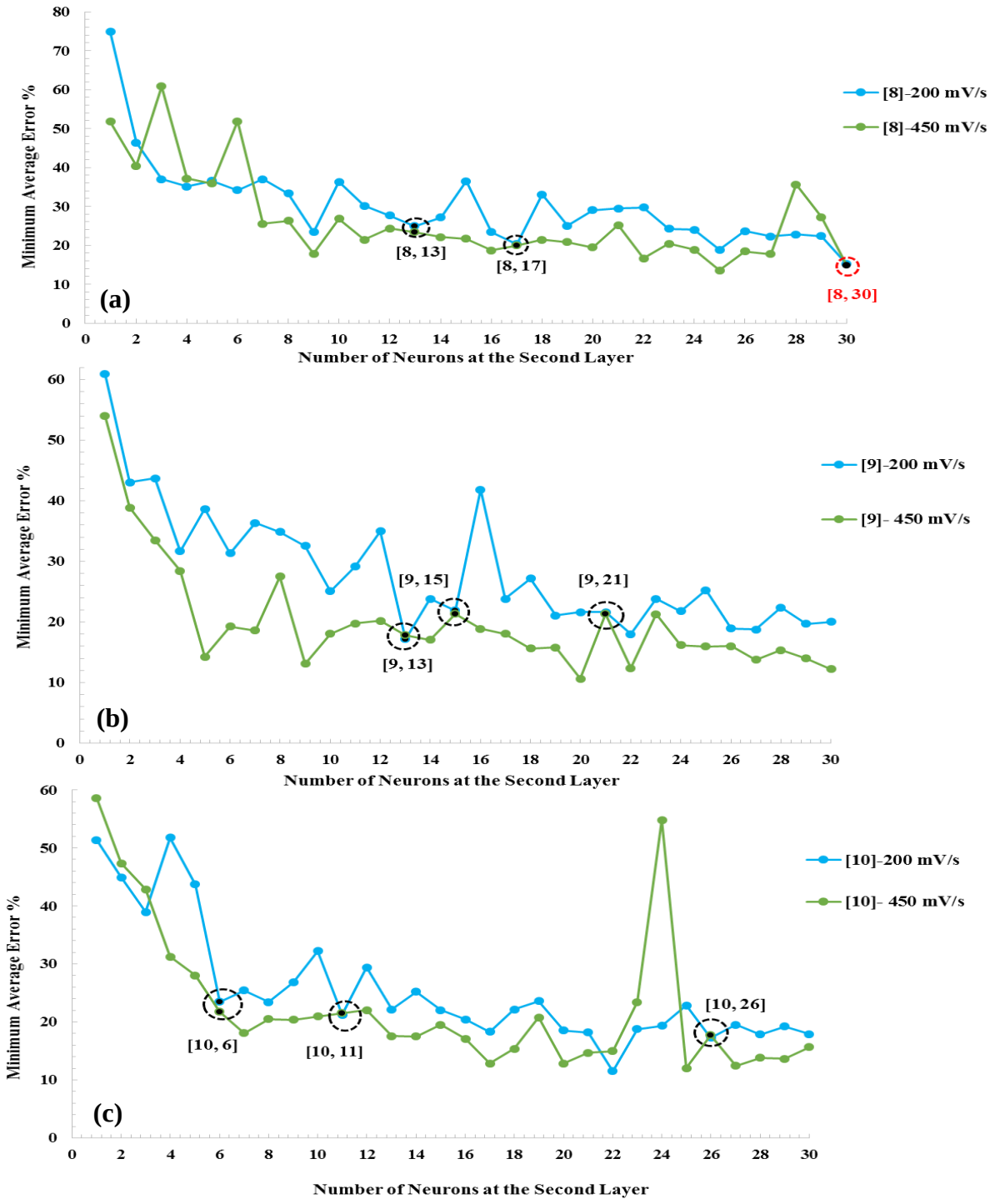

Fig. 4.13 Two hidden layers with 1 to 30 neurons and 1 to 30 validation checks for for 0.5 wt%. at 200 mV/s and 450 mV/s in three structures; (a) [8, 1-30], (b) [9, 1-30], and (c) [10, 1-30] (Black circle = short simulation time; Red circle = long simulation time).

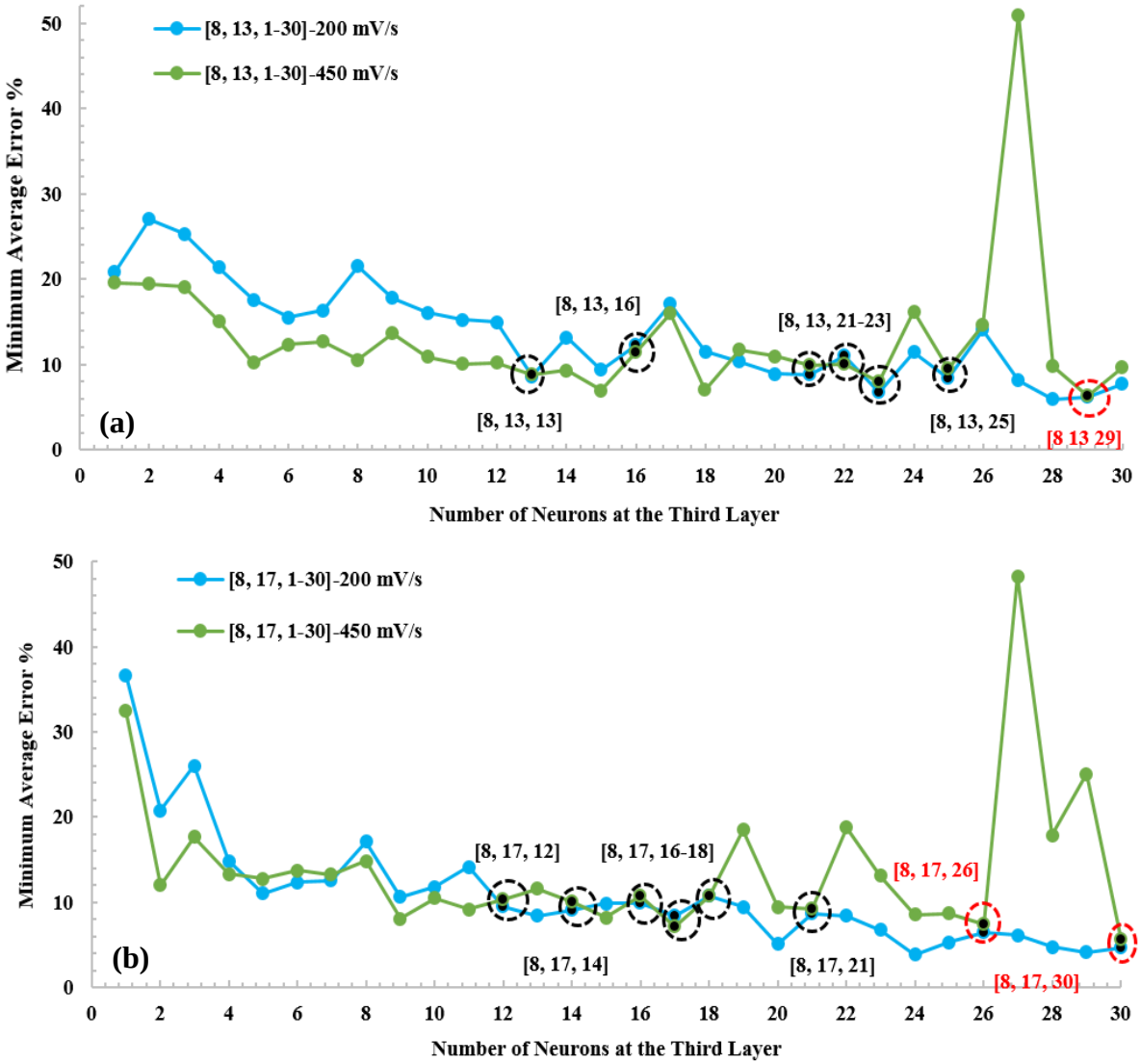

Fig. 4.14 Two hidden layers with 1 to 30 neurons and 1 to 30 validation checks for for 0.5 wt% at 200 mV/s and 450 mV/s in three structures; (a) [8, 13, 1-30], and (b) [8, 17, 1-30] (Black circle = short simulation time; Red circle = long simulation time).

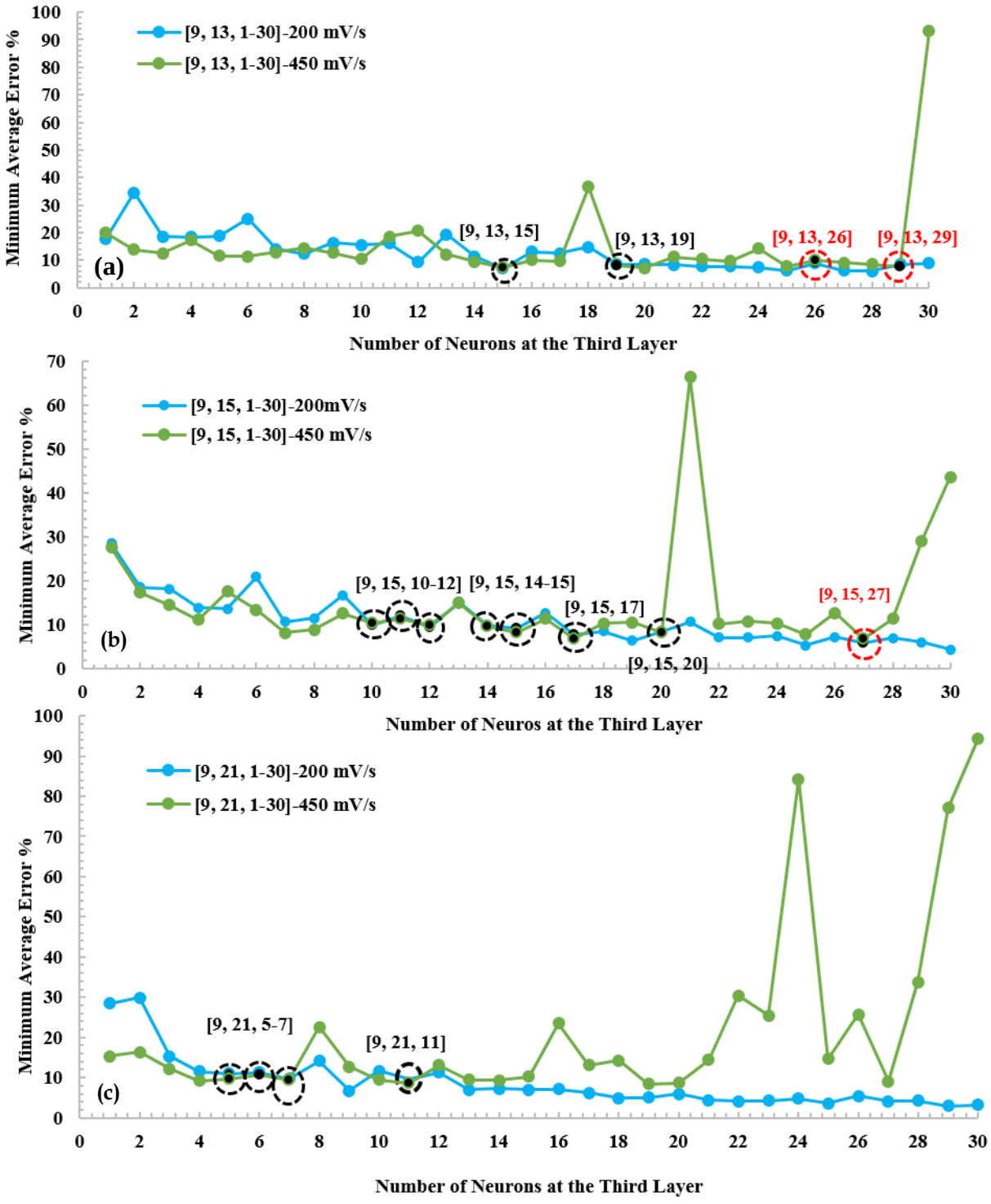

Fig. 4.15 Minimum average error percent for 0.5 wt% at 200 mV/s and 450 mV/s in (a) [9, 13, 1-30], (b) [9, 15, 1-30], (c) [9, 21, 1-30].

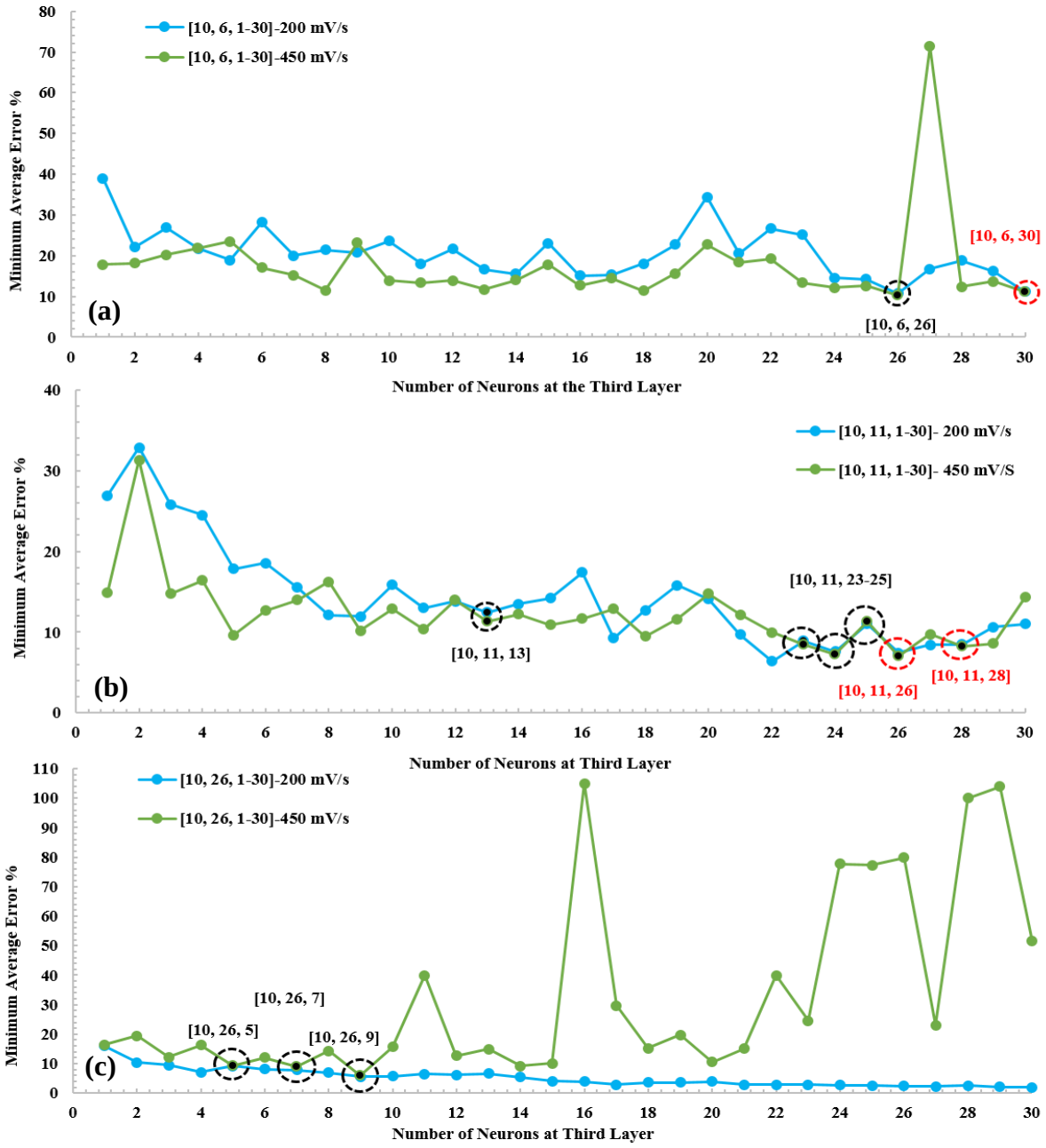

Fig. 4.16 Minimum average error percent for 0.5 wt% at 200 mV/s and 450 mV/s in (a) [10, 6, 1-30], (b) [10, 11, 1-30], and (c) [10, 26, 1-30].

This approach was applied for all final results (black circles) in Figs. 4.13 to 4.16 (all values are listed in Table 4.2). The structures that provide at most 11% error for both 200 and 450 mV/s cases with a difference about 1% are underlined and bolded in Table 4.2. The final results satisfying the mentioned criteria are as follows: [9, 15, 10]-18, [9, 21, 7]-27, [10, 11, 25]-19, and [10, 26, 7]-20. We will refer to these results as the 'final structure [a, b, c]-d' where a, b, and c are the number of neurons in each layer, and d is the number of validation checks.

Table 4.2 Final results related to Figs. 4.13 to 4.16.

	200 mV/s Min Ave Error %	450 mV/s Min Ave Error %	Validation Checks		200 mV/s Min Ave Error %	450 mV/s Min Ave Error %	Validation Checks
[8, 13, 13]	12	8	16	[9, 15, 11]	12	15	12
[8, 13, 16]	12.29	20.58	11	[9, 15, 14]	9.8	49	17
[8, 13, 21]	16.44	9.82	27	[9, 15, 15]	15	8	16
[8, 13, 22]	12.86	10.05	5	[9, 15, 17]	7.80	12	7
[8, 13, 25]	16.16	9.52	9	[9, 15, 20]	8.76	12.22	7
[8, 17, 7]	12.50	14.74	16	[9, 21, 5]	10.85	44	20
[8, 17, 12]	9	14	28	[9, 21, 6]	28.86	10.74	29
[8, 17, 14]	22.43	10	21	**[9, 21, 7]**	**9.72**	**9.32**	**27**
[8, 17, 16]	13.76	10.69	23	[9, 21, 11]	12.97	8.58	19
[8, 17, 17]	8.35	12.22	15	[10, 11, 13]	12.43	12.7	18
[8, 17, 18]	10.74	21.15	15	[10, 11, 23]	9	16	26
[8, 17, 21]	41.52	9.21	17	**[10, 11, 25]**	**11**	**11**	**19**
[9, 13, 15]	6.70	12	19	[10, 26, 5]	18.16	9.24	17
[9, 13, 19]	8.45	11.90	30	**[10, 26, 7]**	**7.84**	**8.81**	**20**
[9, 15, 10]	**9.94**	**10.30**	**18**	[10, 26, 9]	26.69	6.04	27

The predicted results are not the same by repeating one structure because of randomly selected weights and biases by the computer. Therefore, each four final structures were repeated 12 times to compare the predicted values errors for the test sample (0.5 wt%, 450 mV/s). Fig. 4.17 shows that the root mean square error (RMSE) values for predicted out comes with structure [9, 15, 10]-18 are consistently maintaining at the same range in comparison to other structures. The average RMSE values of 12 runs for test sample illustrated in Fig. 4.17 with four mentioned structures are within 0.004 and 0.081. This amount for train samples (0.5 wt% at 200 mV/s) is from 0.0020 to 0.0032. To prove the final results, the CV should be compared to actual experimental data sets.

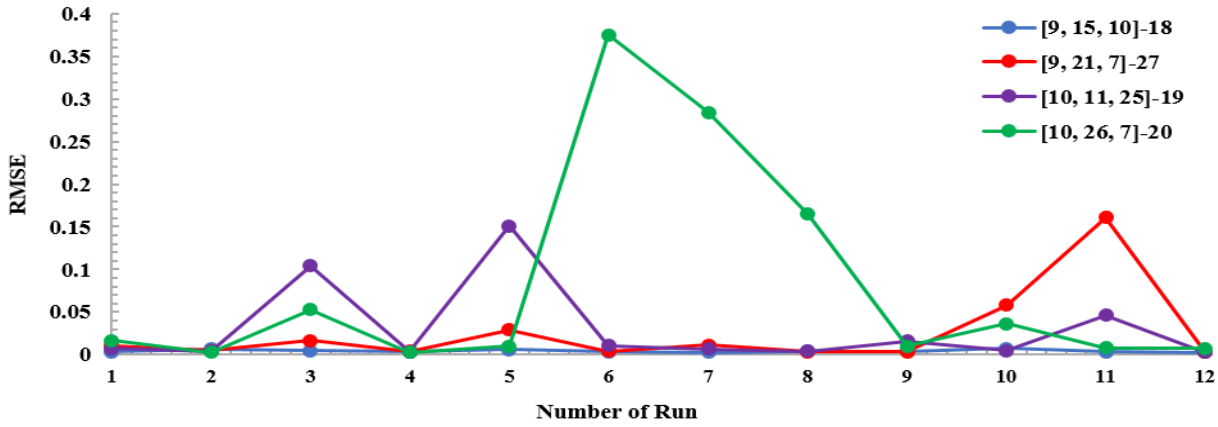

Fig. 4.17 RMSE of test sample for four final structures with 12 runs.

4.2.2.2 CV Comparison

The CV plots of the four final structures with three hidden layers are being compared with experimental data sets (Figs. 4.18 to 4.22) based on the discussion in the previous section. Two distinctive colors are used to distinguish the experimental data collected by Ref. 6 (blue line) and the ANI prediction (red dash line). Figs. 4.18 and 4.19 illustrate the comparison of four cases for

train and test samples. Here, it can be seen that simulated CV curves based from all final four structures capture unique features of both train and test conditions well. In addition, different concentrations and scan rates were also explored to illustrate ANI's predictability and limitation. For this purpose, simulated CV curves for 1 wt% $ZrCl_4$ at 300 mV/s, 2.5 wt% $ZrCl_4$ at 400 mV/s, and 5 wt% $ZrCl_4$ at 250 mV/s are superimposed on the actual experimental data, as shown in Figs. 4.13, 4.14, and 4.15, respectively. Fig. 4.19 shows that the ANI simulation can capture the important features of the CV graph such as oxidation and reduction peaks very well; few deviations can be seen during the transition from the cathodic sweep to anodic sweep region. Fig. 4.20(a) shows comparison between ANI and the diffusion model (green line) as well—a reverse-engineering program design at anodic peak (Ref. 63). Here, it can be seen that the diffusion model indicates a narrow plot coverage. In addition, when the potential is scanned in negative direction, the CV goes far from the experimental data at approximately -1V revealing a limitation of the simple diffusion model. Therefore, this study provides a good prediction and displays the whole trend of CV with a low error. Here, Figs. 4.21 (b) and 4.22 (b) also represent the CV plots with [9, 21, 7]-27 structure indicating that there is a slight difficulty in capturing the cathodic peak in higher concentrations.

Repeatability and distribution of predicted values are very important. Based on the results shown in Fig. 4.12, the best repeatable structure belongs to [9, 15, 10]-18. Thus, to prove this observation, RMSE values for Figs. 4.19 to 4.22 are compared and listed in Table 4.3 revealing that the structure that provides the minimum average RMSE for all tested conditions is related to [9, 15, 10]-18 structure. The next best structure belongs to [10, 11, 25]-19 which shows the average RMSE of 0.0209.

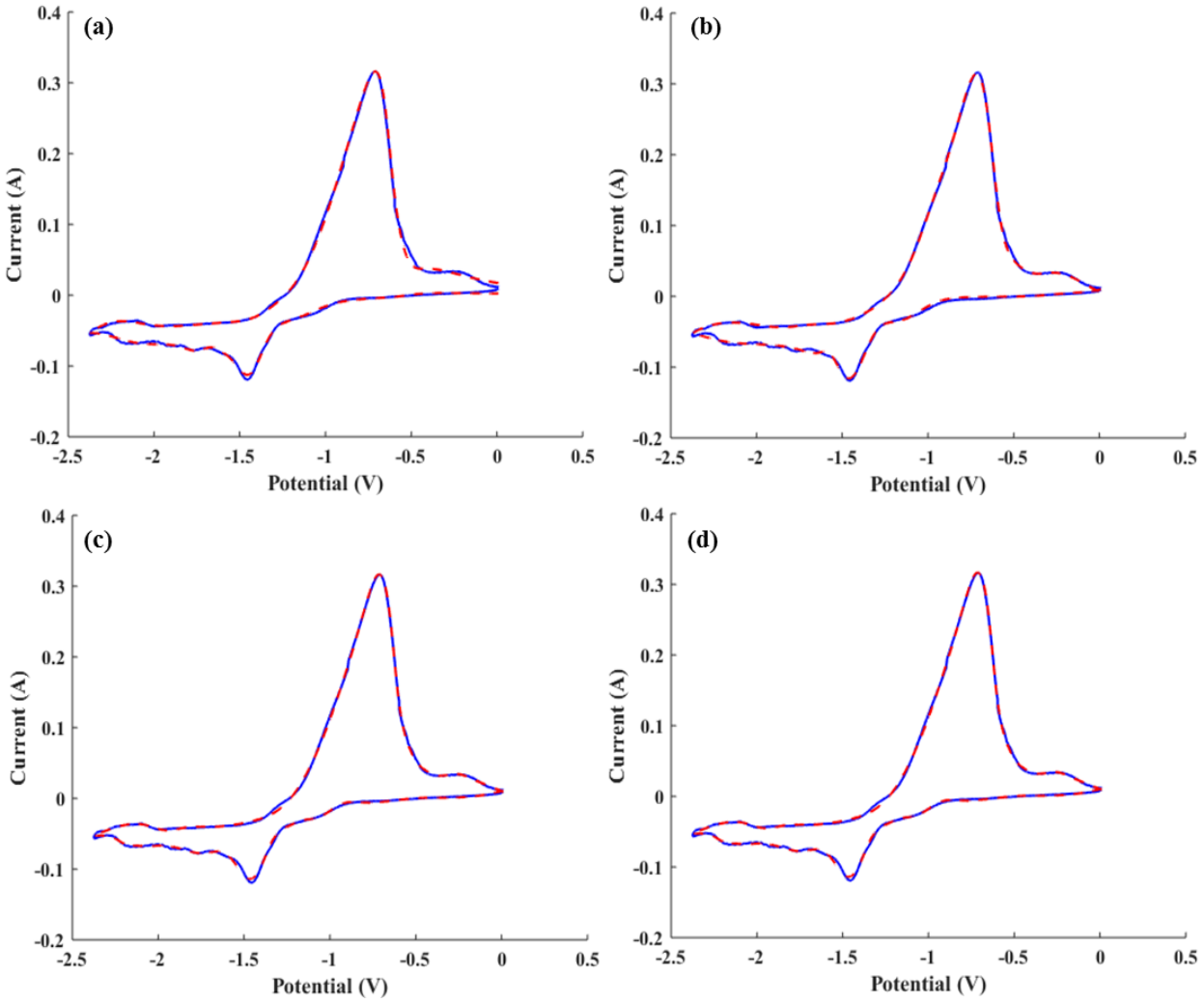

Fig. 4.18 Comparison of CV plot for 0.5 wt% $ZrCl_4$ at 200 mV/s, (a): [9, 15, 10]-18 (b): [9, 21, 7]-27, (c): [10, 11, 25]-19, (d): [10, 26, 7]-20 (Blue line= experimental data, Red dash line =ANI prediction).

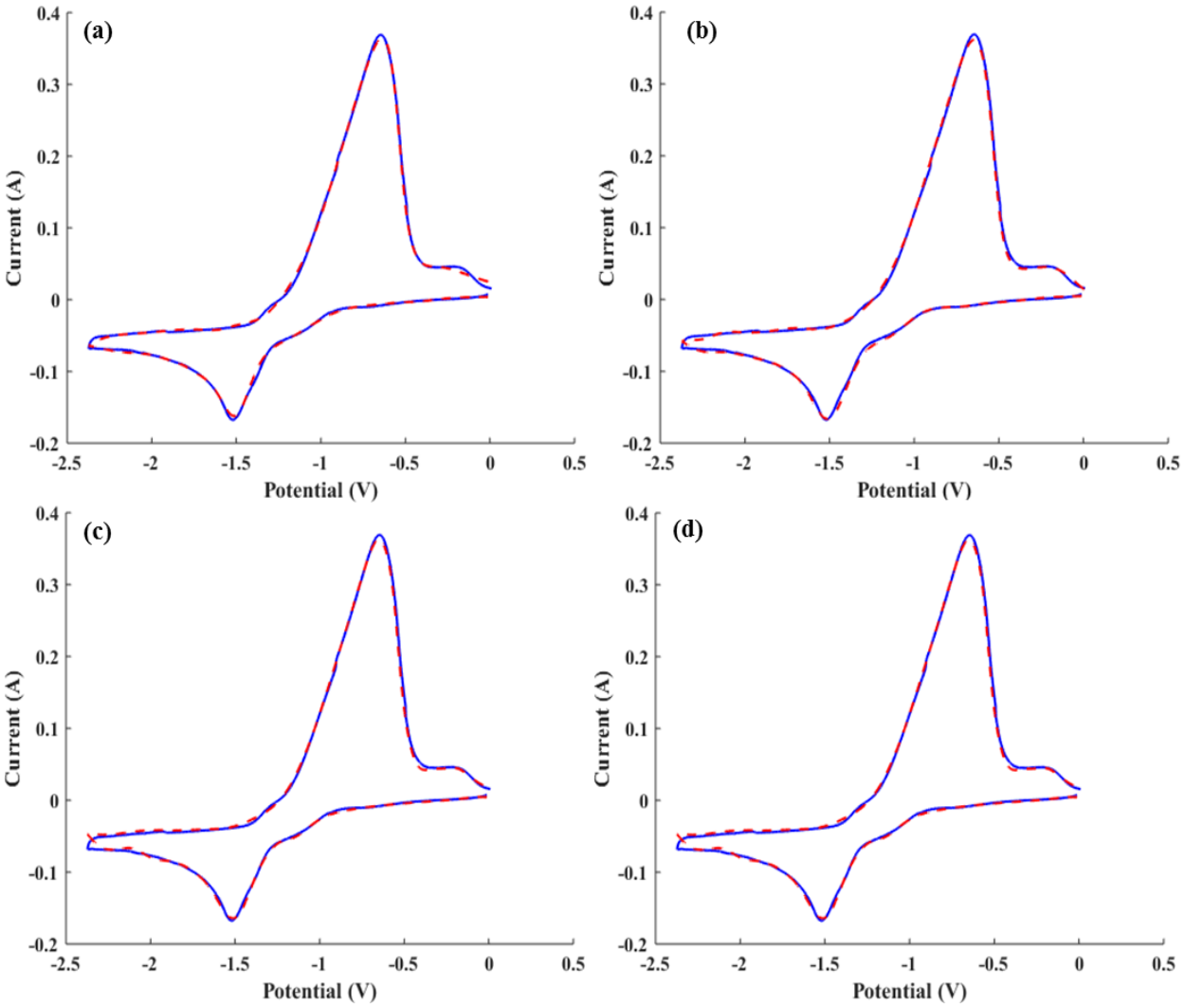

Fig. 4.19 Comparison of CV plot for 0.5 wt% $ZrCl_4$ at 450 mV/s, (a): [9, 15, 10]-18 (b): [9, 21, 7]-27, (c): [10, 11, 25]-19, (d): [10, 26, 7]-20 (Blue line= experimental data, Red dash line =ANI prediction).

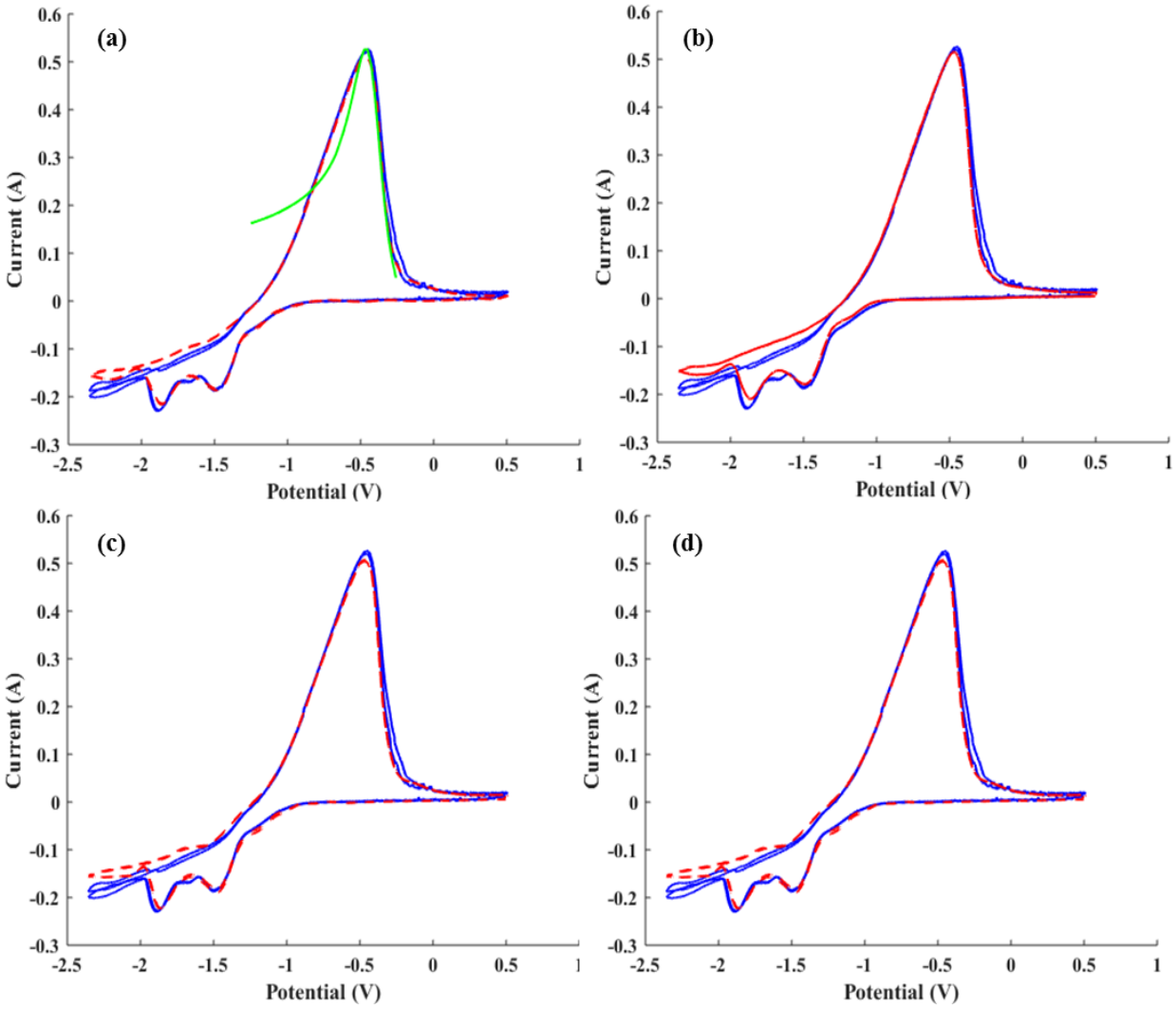

Fig. 4.20 Comparison of CV plot for 1wt% $ZrCl_4$ at 300 mV/s, (a): [9, 15, 10]-18 (b): [9, 21, 7]-27, (c): [10, 11, 25]-19, (d): [10, 26, 7]-20 (Blue line= experimental data, Red dash line =ANI prediction, Green line= Diffusion model).

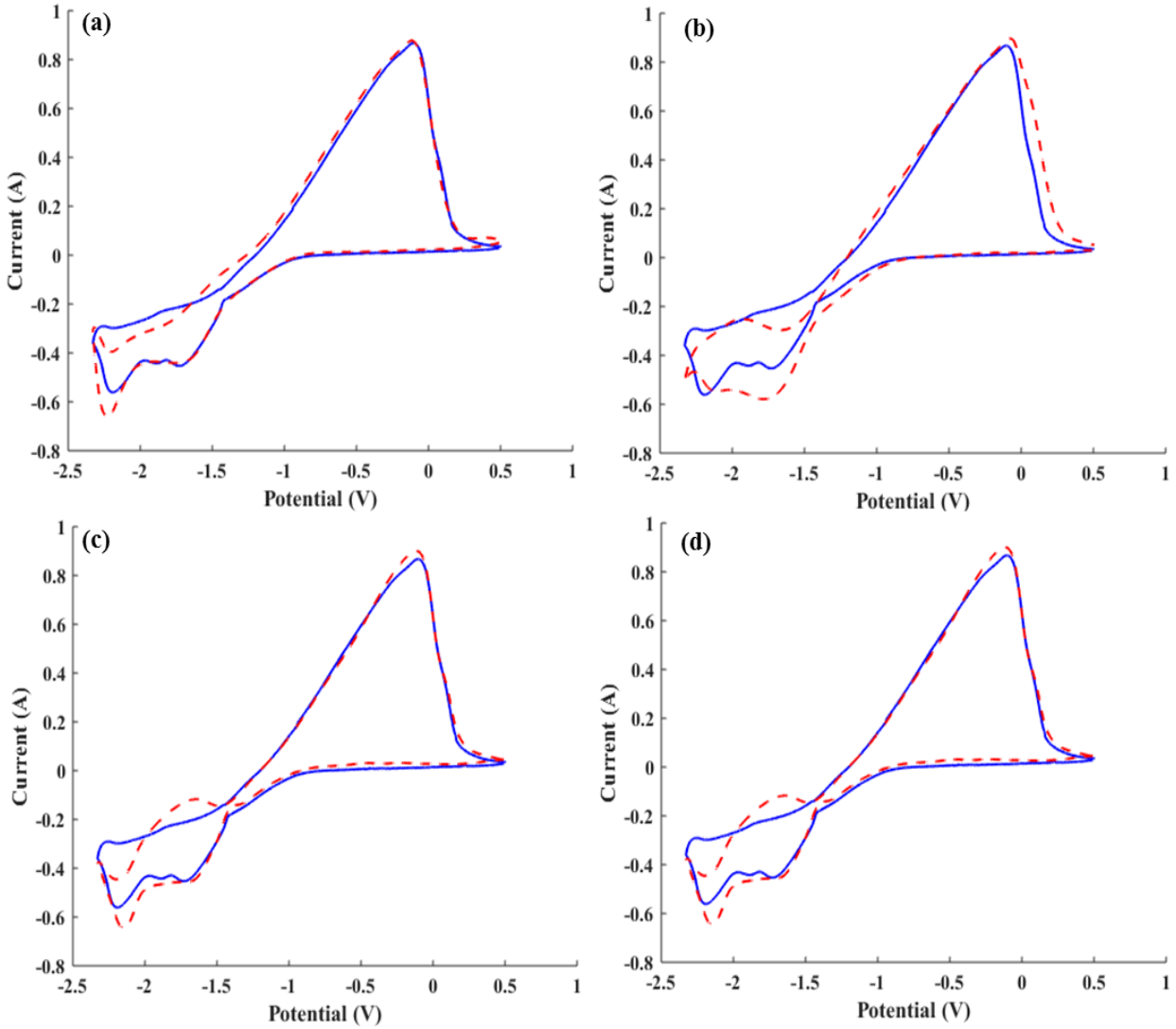

Fig. 4.21 Comparison of CV plot for 2.5 wt% $ZrCl_4$ at 400 mV/s, (a): [9, 15, 10]-18 (b): [9, 21, 7]-27, (c): [10, 11, 25]-19, (d): [10, 26, 7]-20 (Blue line= experimental data, Red dash line =ANI prediction).

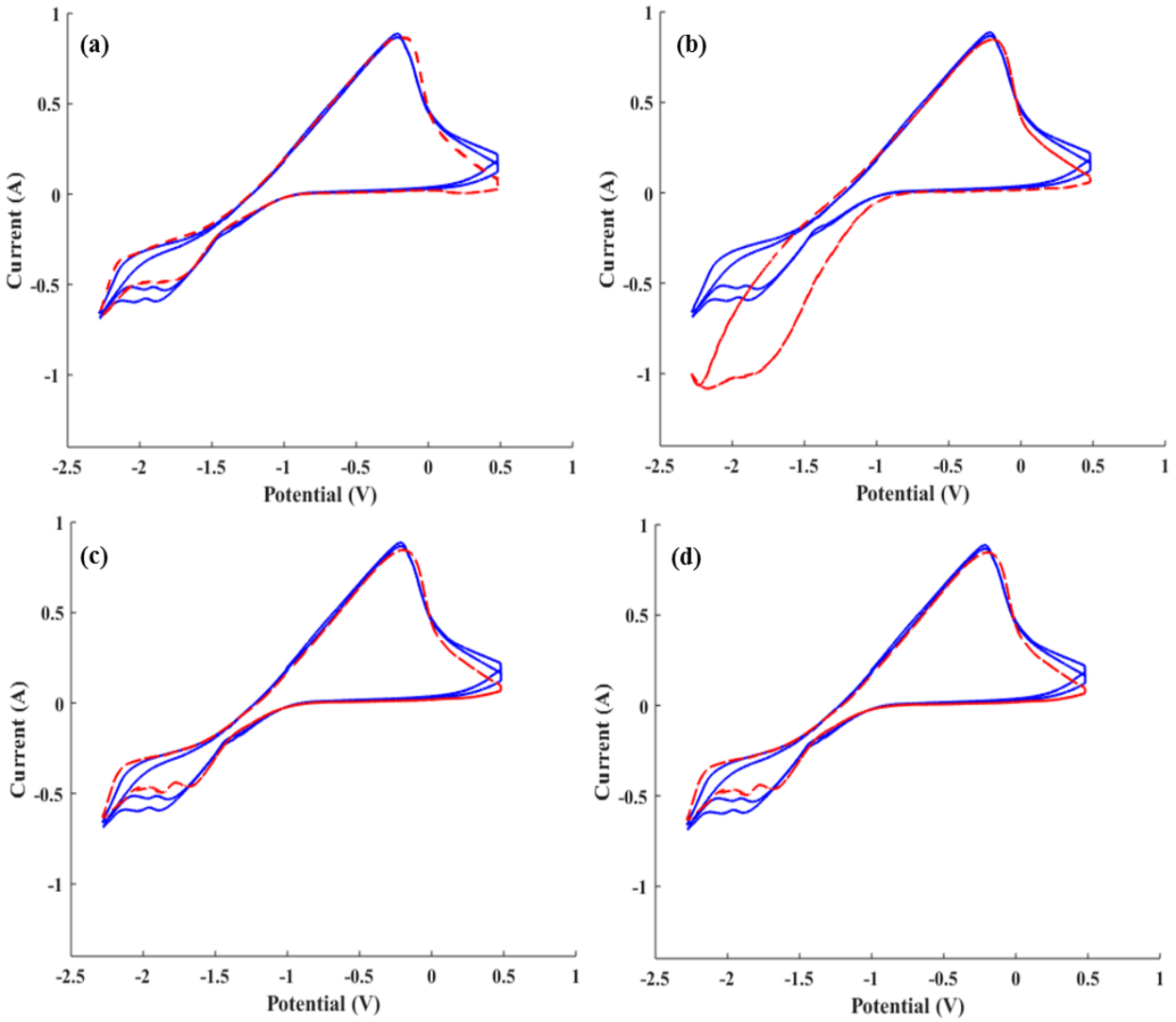

Fig. 4.22 Comparison of CV plot for 5wt% $ZrCl_4$ at 250 mV/s, (a): [9, 15, 10]-18 (b): [9, 21, 7]-27, (c): [10, 11, 25]-19, (d): [10, 26, 7]-20 (Blue line= experimental data, Red dash line =ANI prediction).

Table 4.3 RMSE for Figs. 4.18 to 4.22.

Weight Percent (wt%), Scan Rate (mV/s)	(a): [9, 15, 10]-18	(b): [9, 21, 7]-27	(c): [10, 11, 25]-19	(d): [10, 26, 7]-20
0.5, 450	0.0030	0.0033	0.0036	0.0070
1, 300	0.0129	0.0171	0.0149	0.0160
2.5, 400	0.0346	0.0625	0.0402	0.0524
5, 250	0.0537	0.2142	0.0544	0.0646

4.3 Uranium Chloride

4.3.1 Computational Procedure

In this section, the two final structures of zirconium have been applied for 5, 7.5, and 10 wt% of uranium chloride (UCl_3) in LiCl-KCl eutectic molten salt at 773 K under various scan rates (giving a total of 350,000 data points) to prove the ANI compatibility and concept. These two structures, [9, 15, 10]-18 and [10, 11, 25]-19, are denoted as "structure-1", and structure-2" in this section. Three fixed training data set combinations have been considered and reported in Table 4.4, denoted by Sections (a), (b), and (c). The training data sets are highlighted and the test data sets are indicated in white. Here, two conditions of 5 wt% at 100 mV/s and 450 mV/s are considered as train and test samples. From Table 4.4, the total training data set for Section (a) is ~49% of experimental data sets. This value for Sections (b) and (c) are ~49%, and ~51%, respectively. As it is mentioned in Section 4.2, the ANI prediction will be changed by repeating one structures due to the randomly selected weights and biases by the computer. Each fixed combination with structure-1 and structure-2 has been repeated 10 times to prove the repeatability and distribution of predicted values. Figs. 4.23 and 4.24 illustrate the RMSE of Sections (a) to (c) with structure-1 and 2, respectively. Fig. 4.23 shows that the RMSE values for the predicted outcomes with Sections (a) and (b) are being maintained approximately at the same range after the 4^{th} run, in comparison to Section (c). This outcome indicates that by increasing the number of training data sets it does not necessarily help improving the the prediction process (this may also result in over training). Fig. 4.24 reveals that the repetition results with structure-2 do not follow any specific pattern; that is, the prediction occurs randomly.

Table 4.4 Different test and training data set combinations of experimental data sets for UCl_3 in LiCl-KCl at 773K.

	(mol/cm^3)	Scan Rate (mV/s)									
(a)	5wt%	100 Train	150 Test	200 Train	250 Test	300 Test	350 Train	400 Test	450 Test	500 Train	600 Test
		700 Test	800 Train	900 Test	1000 Train	2000 Train					
	7.5 wt%	200 Train	250 Test	300 Train	350 Test	400 Test	450 Test	500 Train	600 Test	700 Train	800 Test
		900 Train	1000 Test	1100 Test	1200 Train	1300 Test	1400 Test	1500 Train	1600 Test	1800 Train	2000 Train
	10 wt%	200 Train	450 Train	500 Train	600 Test	700 Test	800 Test	900 Train	900 Test	1000 Test	1100 Train
		1200 Test	1300 Test	1400 Test	1500 Train	1600 Test	1700 Test	1800 Train	1900 Test	2000 Train	2500 Train
		2500 Test	3000 Train	3500 Test	4000 Train						
(b)	(mol/cm^3)	Scan Rate (mV/s)									
	5wt%	100 Train	150 Train	200 Test	250 Train	300 Test	350 Test	400 Train	450 Test	500 Test	600 Train
		700 Test	800 Train	900 Test	1000 Train	2000 Train					
	7.5 wt%	200 Train	250 Train	300 Test	350 Test	400 Train	450 Test	500 Train	600 Test	700 Train	800 Test
		900 Train	1000 Test	1100 Test	1200 Train	1300 Test	1400 Test	1500 Train	1600 Test	1800 Train	2000 Train
	10 wt%	200 Train	450 Train	500 Test	600 Test	700 Train	800 Test	900 Test	900 Test	1000 Train	1100 Test
		1200 Test	1300 Train	1400 Test	1500 Test	1600 Train	1700 Test	1800 Train	1900 Test	2000 Train	2500 Train
		2500 Test	3000 Train	3500 Test	4000 Train						
(c)	(mol/cm^3)	Scan Rate (mV/s)									
	5wt%	100 Train	150 Test	200 Train	250 Test	300 Train	350 Test	400 Train	450 Test	500 Test	600 Train
		700 Test	800 Train	900 Test	1000 Train	2000 Train					
	7.5 wt%	200 Train	250 Train	300 Test	350 Test	400 Train	450 Test	500 Train	600 Test	700 Train	800 Test
		900 Train	1000 Test	1100 Test	1200 Train	1300 Test	1400 Test	1500 Train	1600 Test	1800 Train	2000 Train
	10 wt%	200 Train	450 Train	500 Test	600 Test	700 Train	800 Test	900 Train	900 Test	1000 Train	1100 Test
		1200 Test	1300 Train	1400 Test	1500 Test	1600 Train	1700 Test	1800 Train	1900 Test	2000 Train	2500 Train
		2500 Test	3000 Train	3500 Test	4000 Train						

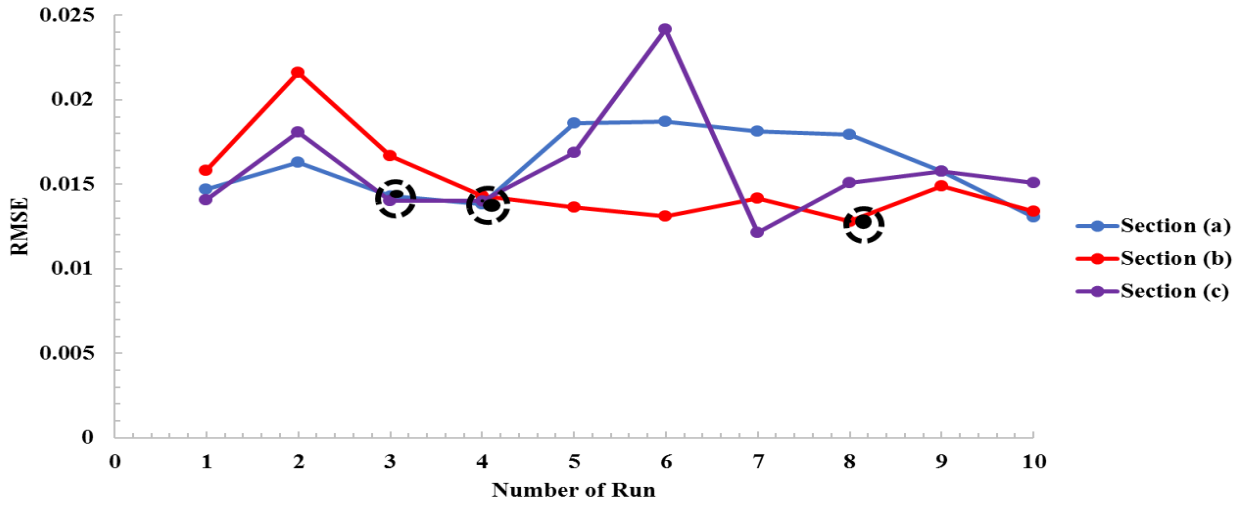

Fig. 4.23 RMSE of test sample for [9, 15, 10]-18 structures with 10 runs.

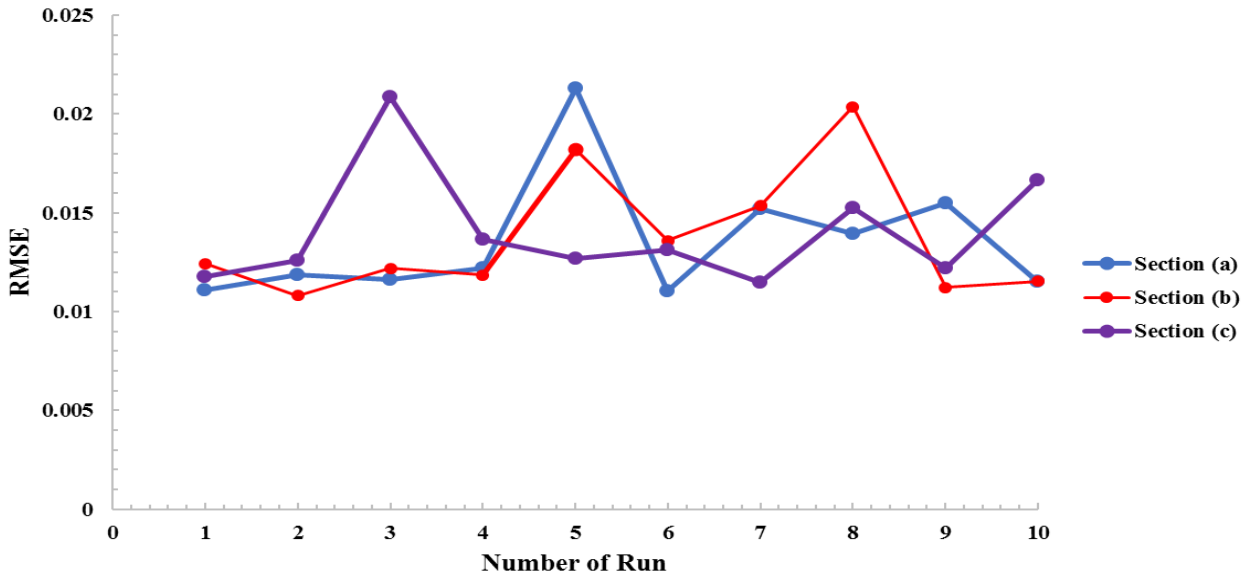

Fig. 4.24 RMSE of test sample for [10, 11, 25]-19 structures with 10 runs.

The process time is also a critical factor to select a desirable structure. Thus, to choose one combination and one structure, the RMSE for all concentrations must be considered. The average RMSE and process time of 10 run for Table 4.4, Sections (a) to (c) with both structure-1 and 2 are listed in Table 4.5. From this table, the minimum average process time and RMSE are related to

Section (a) with structure-1. One run from Table 4.4, Sections (a) to (c) with structure-1 has been selected as a better outcome by concentrating on minimum error prediction of the total test data set and are marked by the dashed circles in Fig. 4.23. To prove the final results, the CV plots of Table 4.4, Sections (a) to (c) with structure-1 should be compared to actual experimental data sets.

Table 4.5 Average process time and RMSE for Table 4.4, Sections (a) to (c) related to selected run indicated in Figs. 4.23, and 4.24.

	Average Process Time (Minute)		**Average RMSE**	
	Structure-1	Structure-2	Structure-1	Structure-2
Section (a)	10	14	0.0796	0.0872
Section (b)	14	28	0.1221	0.1397
Section (c)	22	27	0.1330	0.1218

4.3.2 CV Comparison

In this section, the simulated CV curves from Table 4.4, Sections (a) to (c) with structure-1 for the selected runs at different concentrations and scan rates are being compared with experimental data sets (Figs. 4.25 to 4.29). Two distinctive colors are used to distinguish the experimental data collected from Ref. 6 (blue line) and the ANI prediction (red line). Although the ANI simulation for all three sections with structure-1 can capture the main objective of CV curves, Section (a) provides a better prediction in different concentrations and scan rates. It is important to mention that Figs. 4.25 and 4.26 are related to train and test samples, respectively. Furthermore, the Figs. 4.27 to 4.29 are related to a common tested conditions for three combinations. Based on the results shown in Fig. 4.23, the best repeatable combination belongs to structure-1. Thus, to prove this observation, RMSE of predicted values for Figs. 4.27 to 4.29 are

compared in Table 4.6. The result displays that the structure provides the minimum average RMSE for most of the conditions is related to structure-1 indeed.

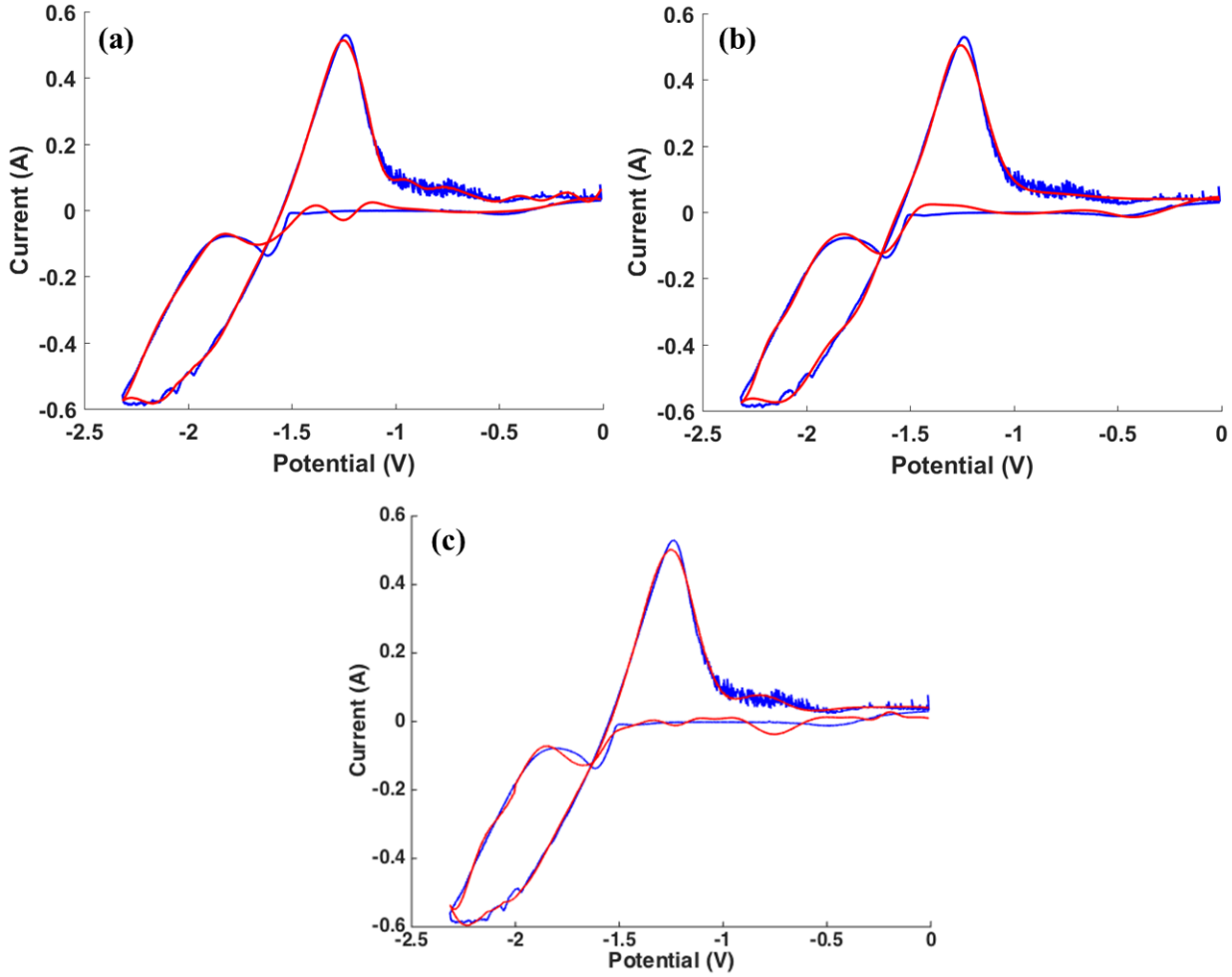

Fig. 4.25 Comparison of CV plot for 5wt% UCl_3 at 100 mV/s with structure-1, (a): Section (a), (b): Section (b), (c): Section (c) combination (Blue line= experimental data, Red dash line =ANI prediction).

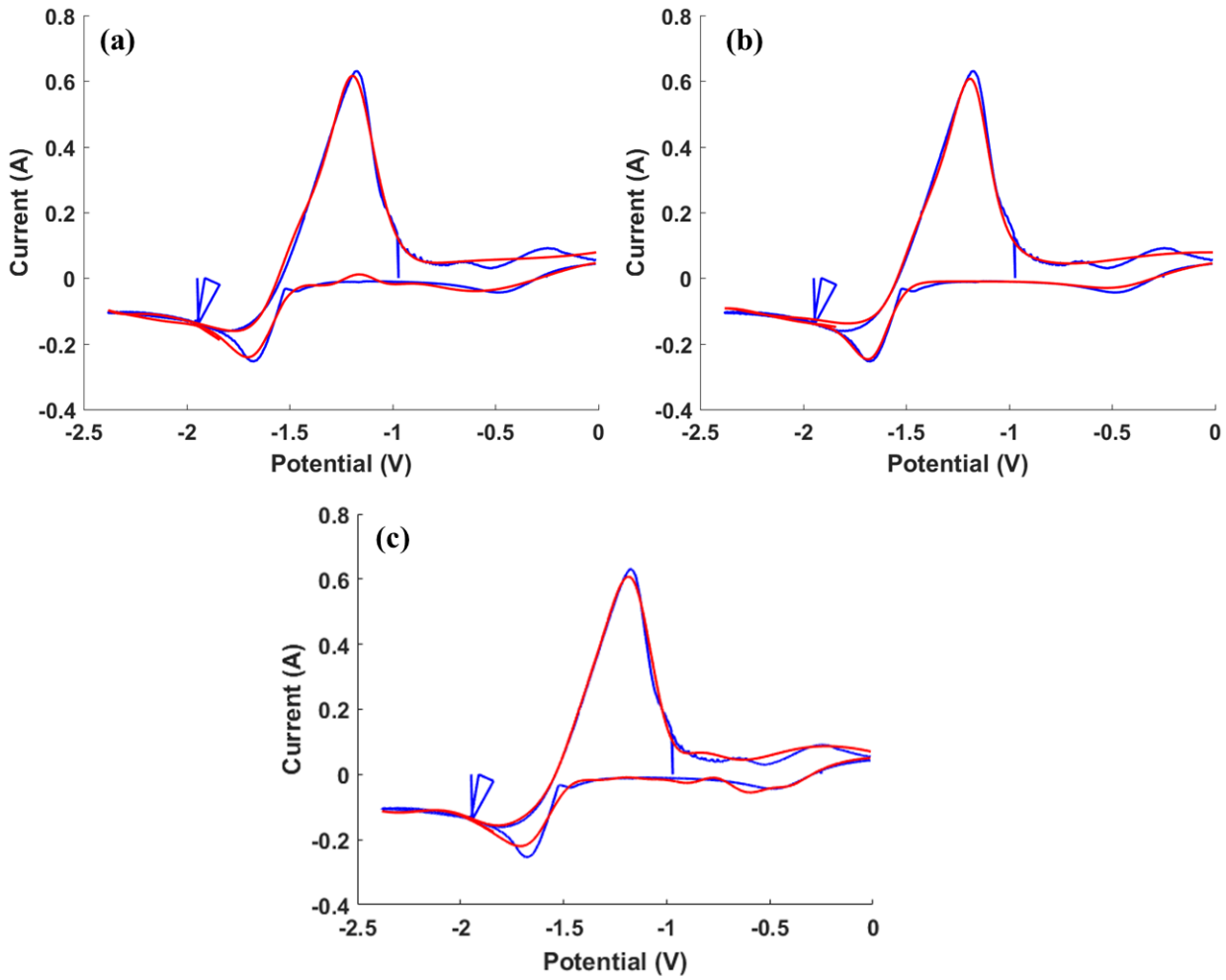

Fig. 4.26 Comparison of CV plot for 5wt% UCl_3 at 450 mV/s with structure-1, (a): Section (a), (b): Section (b), (c): Section (c) combination (Blue line= experimental data, Red dash line =ANI prediction).

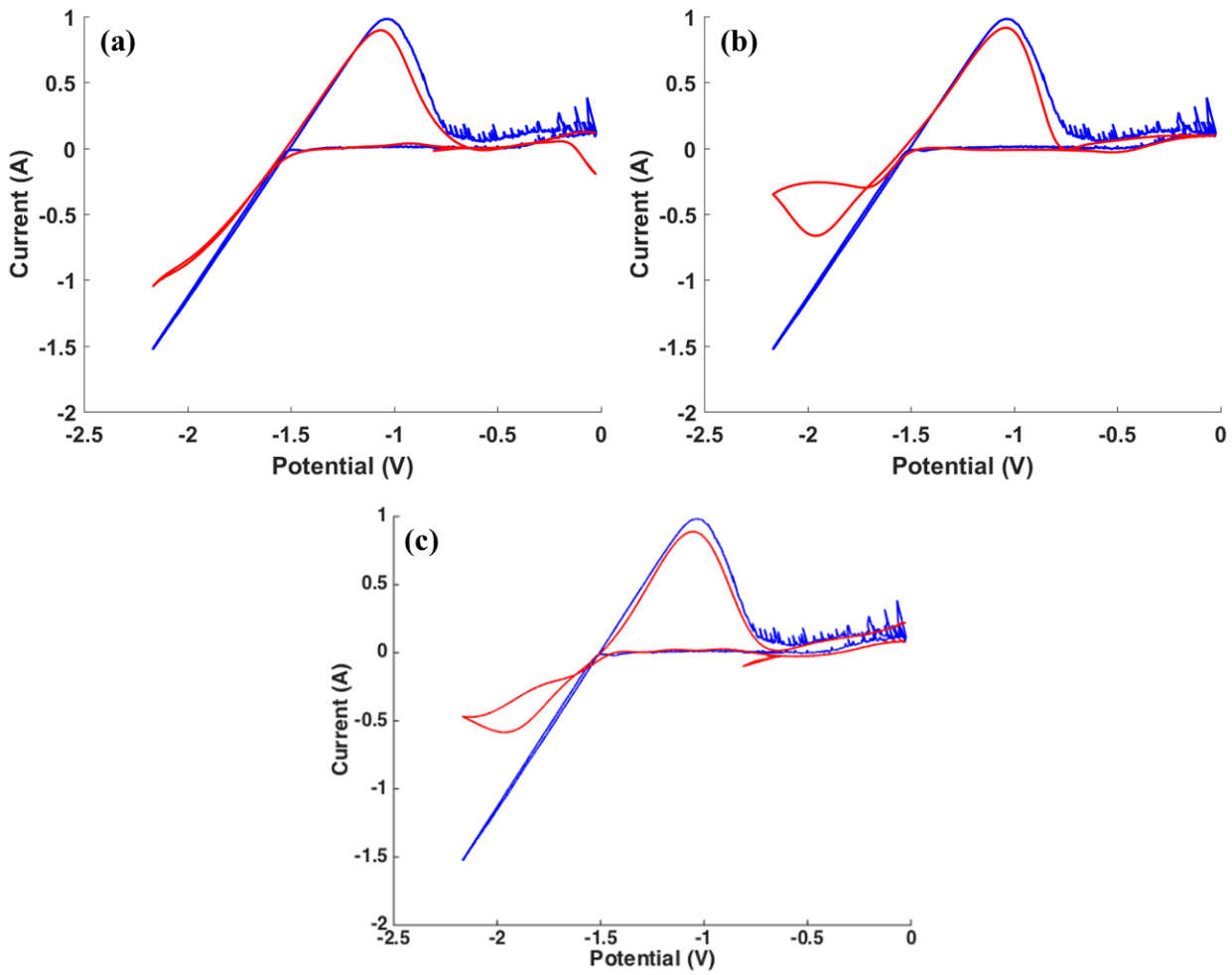

Fig. 4.27 Comparison of CV plot for 7.5wt% UCl_3 at 350 mV/s with structure-1, (a): Section (a), (b): Section (b), (c): Section (c) combination (Blue line= experimental data, Red dash line =ANI prediction).

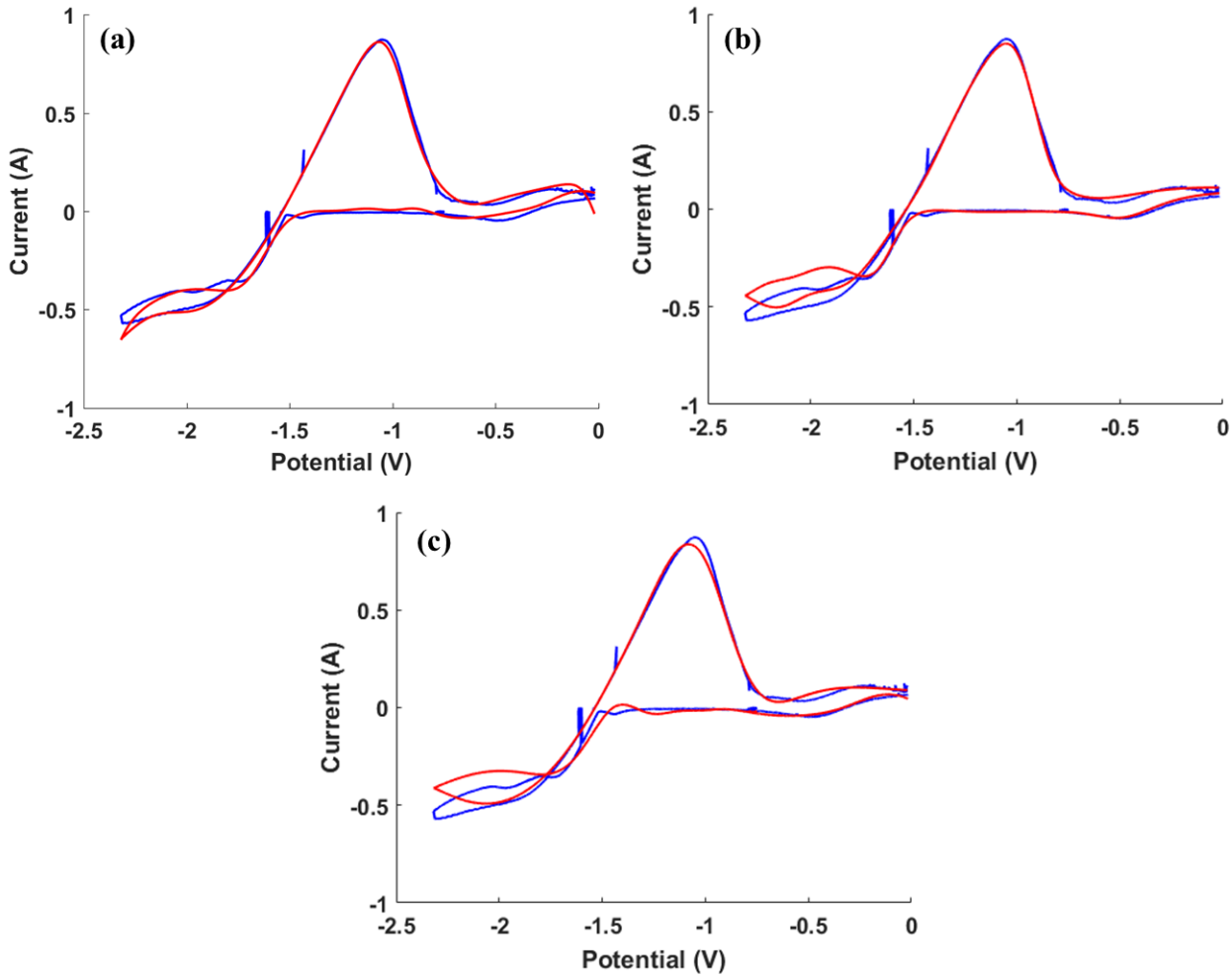

Fig. 4.28 Comparison of CV plot for 7.5wt% UCl_3 at 450 mV/s with structure-1, (a): Section (a), (b): Section (b), (c): Section (c) combination (Blue line= experimental data, Red dash line =ANI prediction).

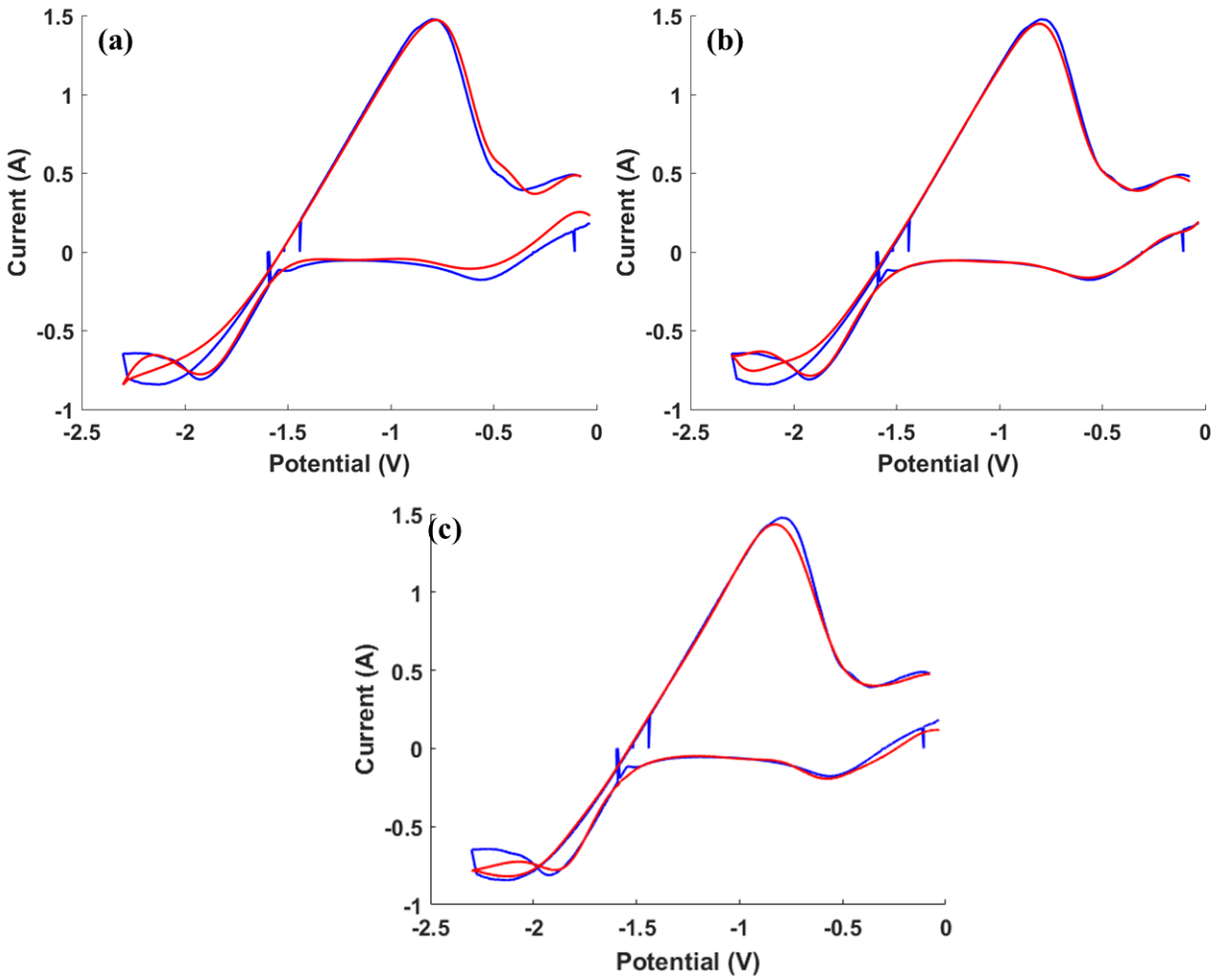

Fig. 4.29 Comparison of CV plot for 10wt% UCl_3 at 1700 mV/s with structure-1, (a): Section (a), (b): Section (b), (c): Section (c) combination (Blue line= experimental data, Red dash line =ANI prediction).

Table 4.6 RMSE for Figs. 4.26 to 4.29.

Weight Percent (wt%), Scan Rate (mV/s)	Section (a)	Section (b)	Section (c)
5, 450	0.0143	0.0128	0.0140
7.5, 350	0.1505	0.3478	0.3199
7.5, 450	0.0249	0.0362	0.0395
10, 1700	0.1586	0.3654	0.3385

4.4 Conclusion

We presented a study of data analysis with ANI for the electrorefiner used in pyroprocessing technology. We analyzed zirconium chloride concentrations of 0.5, 1, 2.5, and 5 wt% at different scan rates at 773 K based on the experimental data set of Refs. 6 and 9 to illustrate the ANI ability of handling a complex system. The minimum input data that can be considered as training data is 43% of over 230,000 experimental data points. One, two, and three hidden layers with 1 to 30 neurons at each layer, and 1 to 30 validation checks were analyzed. The minimum average percent errors for train and test samples were calculated. The work shown here proved that a framework for applying ANI could be utilized to bypass the guessing approach and omit trial and error method. Therefore, the system was able to stop at a reasonable point without going beyond underfitting and overfitting. The results demonstrate that adding hidden layers for a fix training data set results in a smaller learning (modelling) error. The criteria for defining first hidden layer entailed test and train sample which provided the average percent error less than 60% with difference around 5%. For two hidden layers, this scale was tuned to 25% with difference below 2% and for three hidden layers; it is limited to 12% and 1.2%. The average RMSE values of 12 runs for test sample illustrated in Fig. 4.17 with four mentioned structures can be fallen in 0.004 to 0.081. This amount for train samples (0.5 wt% at 200 mV/s) is from 0.002 to 0.0032. Two selected structures shown more productive predictions are related to [9, 15, 10]-18 and [10, 11, 25]-19.

To prove the ability of ANI concept on another chloride salt, two final structures from zirconium chloride study were applied to uranium chloride of 5 to 10 wt% in LiCl-KCl eutectic

molten salt at 773 K (above 350,000 points). In this study, three different fixed data combinations were considered. Each combination with two structures was repeated 10 times. The summary of the matrix can be found in Table 4.4, Section (a) with [9, 15, 10]-18 provides a better prediction with the average RMSE around 0.0796 and average process time of ~10 minutes. The results show that different data combinations may provide different results. However, based on the main objective of this part to demonstrate the ANI's ability to predict the complex CV of another component, our resulting outcomes are promising. In conclusion, the ANI implementation can be successfully deployed as an alternative method of robust signal detection towards safeguards application in pyroprocessing technology.

Chapter 5: Summary, Conclusion and Future Work

5.1 Summary

5.1.1 Chapter 1: Purpose, Motivation, Approach

- Pyroprocessing, which was developed by Argonne National Laboratory (ANL), is a high-temperature reprocessing method of EBR-II for UNF.
- ER is the heart of pyroprocessing technology wich contains the dynamic compositions of molten salt to recover pure uranium at the cathodic side.
- The standard material accountancy method used commonly at the national laboratories is the ICP-MS or ICP-OES method, which may take up to 3 – 4 weeks to obtain all material compositions.
- Several methods have been proposed and supported by the DOE-NEUP, including LIBS, UV-VIS, and Electrochemical methods (both experimental and modeling routines).
- Despite successful modeling studies for the ER through the cyclic voltammogram techniques, predicting the trace of species without experimental data sets in a relatively short time has still remained as an issue and become a great need in nuclear material detection and accountancy.
- The goal of this study is to develop a near real time monitoring detection program to trace the trend of each species and predict the unseen situation toward pyroprocessing safeguards.

- For this purpose, a diffusion model has been developed to predict the cyclic voltammetry (CV) of uranium chloride in different scan rates and conditions in a short time.
- To provide a compatible model with complicated material such as zirconium chloride, a novel electrochemical data analysis using an artificial neural intelligent (ANI) method has been proposed and developed.

5.1.2 Chapter 2: Review of Electrochemical Process

- The reprocessing of UNF while contains 96% of uranium is very significant to cut down the volume of radioactive waste and decrease the need for uranium sources.
- Pyroprocessing technology, known as electrochemical process, electrometallurgical reprocessing, or pyrochemial technology, is a dry reprocessing process. This technology is a high-temperature (T > 723 K) method to separate uranium and plutonium from used metallic nuclear fuel of EBR-II.
- The main part of pyroprocessing is the Mark-IV ER:
  - Here, chopped used fuel are entered the ER and the uranium fuel is produced after removing the cadmium or adhered salt through the cathode process and casting furnace.
  - The High Level Wastes (HLWs) are converted to ceramic and metallic waste forms as well.
  - ER is consisted of anodic and cathodic electrode with LiCl-KCl molten salt as the electrolyte.
  - The pure uranium can be recovered by cotrolling the applied voltage.

- One of the most common electroanalytical methods for determining the thermodynamic and electrochemical behavior of elemental species in the eutectic molten salt LiCl-KCl inside ER is cyclic voltammetry (CV)—this method can possibly be used to estimate diffusion coefficients, apparent standard potentials, transfer coefficients, and numbers of electron transferred.
- From CV of UCl_3 in LiCl-KCl molten salt at 773K, as shown in Fig. 2.7, two major cathodic and anodic peaks can be found with the following reactions:

A_c: $$U^{+4} + e^- \rightarrow U^{+3} \quad (5.1)$$

C_c: $$U^{+3} + 3e^- \rightarrow U \quad (5.2)$$

C_a: $$U \rightarrow U^{+3} + 3e^- \quad (5.3)$$

A_a: $$U^{+3} \rightarrow U^{+4} + e^- \quad (5.4)$$

5.1.3 Chapter 3: Diffusion Model

- The diffusion coefficients can be determined using Randles-Sevick equation for reversible side and Delahay equations for irreversible side (Eqs (3.1) and (3.12)) by considering the experimental data of 1 to 10 wt% UCl_3 and 1 to 5 wt% $ZrCl_4$ in LiCl-KCl molten salt operating at 773 K (Ref. 6).
- The apparent standard potential can be related to the diffusion coefficient for a soluble/insoluble irreversible redox couple by the following equation:

$$E^{\circ *} = E_{PC} + \left(\frac{RT}{n\alpha F}\right)\left[0.78 - \ln k_S + \ln\left(\sqrt{\frac{n\alpha F\nu D}{RT}}\right)\right] \quad (5.5)$$

- The current and potential at different time can be calculated by numerically solving the Fick's law with the boundary and initial conditions mentioned in Section 3.2, throuth Laplace transform, variation changes, and convolution theorem. In addition, the surface

concentration of oxidant and reductant species has been calculated for four major regions: (1) the reversible cathodic, (2) the irreversible cathodic, (3) irreversible anodic, and (4) reversible anodic.

- The diffusion model which provides the current versus potential plot and concentration of each species versus time is writen in *Matlab* software package, and GUI. The run time for each concentration and scan rate is less than 2 minute with the time interval around 0.08 second.
- To determine the diffusion coefficient and formal electrode potentials for each concentration and scan rates, the average of diffusion coefficients and apparent standard potentials reported in the literatures are tuned/adjusted with 10^{-7} and 0.0002 interval, respectively. These values would be adjusted by less than 10 iterations.
- Fig. 5.1 illustrates the results of this study showing the important features of the CV graph such as the potential and current information at each peak with the RMSE of potential and current around 0.00764 and 0.0178, respectively.
- Although the model is not able to capture the adsorption peaks and shows a dissimilarity, the main focus of this study has been accomplished showing the ability to capture the anodic and reduction peaks of experimental data sets.

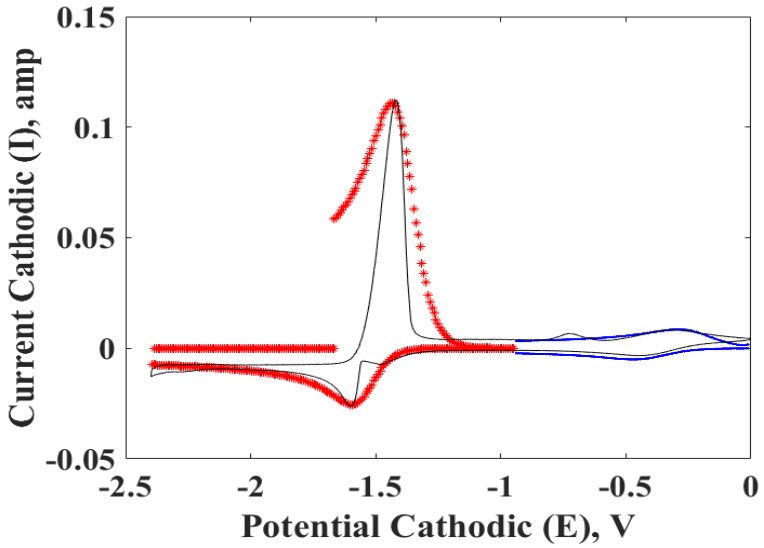

Fig. 5.1 Cyclic voltammograms of 1 wt% UCl_3 with 100 mV/s in LiCl-KCl eutectic at 773 K.

- The developed code is able to calculate the concentration of reduced and oxidized species as a function of time. In addition, the number of desired points can be entered by the user in the *Matlab* code to deliver the concentration of reduction and oxidation species, and the process time at each point.
- The CV of zirconium chloride is complicated in nature due to several mechanistic reactions occurring during the CV run (including obscure diffusion coefficients values at each peak).
    - To determine the diffusion coefficient and apparent standard potential, the initial guess of diffusion coefficient and formal potential for each peak were adjusted between 900 to 1700 iterations.
    - The processing time to simulate cathodic peaks for zirconium was generally lasted about 10 minutes and for highest anodic peak, it could take up to 2 to 3 hours.
- This study concluded that the combination of Zr^{+2}/Zr and Zr^{+4}/ Zr^{+2} is occurring at the first cathodic peak (A_c in Fig. 3.9), Zr^{+2}/Zr at the second (B_c) and Zr^{+4}/Zr at the third peak (C_c).

At the anodic section, the first oxidation reaction (B_a) is related to the combination of 70% Zr/Zr^{+4} and 30% Zr/Zr^{+2} for the 1.07 wt% and 2.49 wt% zirconium chloride and 30% Zr/Zr^{+4} and 70% Zr/Zr^{+2} for 4.98 wt% $ZrCl_4$. The Zr^{+2}/ Zr^{+4} reaction is happening at the second anodic peak (A_a).

5.1.4 Chapter 4: Artificial Neural Intelligent

- A novel signal detection through artificial neural intelligent (ANI) is proposed as an alternative electrochemical method to predict the CV plot. Due to the similarity between computer and brain, a computer has capability of learning by feeding massive input data.
- One of the ANI network which is considered in this study is Multi layer perceptorn (MLP).
    - It is consisted of one input layer, different hidden layers and one output layer, which are interconnected by a number of nodes called neurons.
    - The weighted inputs are sum with a constant bias and enter to the activation function, which is sigmoid function in this study, and giving the outputs.
- Input data are divided into training data set, validation data sets, and test data set:
    - *Training data set* is part of the input data sets for adjusting the weights and bias;
    - *Validation data set* is used to minimize the overtraining; and
    - Whatever is left is related to *test data set* to assess the system performance.
- In this study the output variable is current and the input variables are potential, process time, weight percent, and scan rate related to the uranium chloride with 1 to 10 wt% and zirconium chloride with 5 to 10 wt%. The total experimental for uranium and zirconium are above 350,000 and 230,000 points, respectively.
- The procedure in this work is running ANI on different hidden layers (1 to 3) with various neurons (1 to 30) at several validation checks (1 to 30).

- The average percent errors between experimental and predicted data sets for two selected conditions (one from test data set and one from train data set) are calculated.
- Then, the structure that both cases provide a minimum average percent error is selected.
- Next, the number of the hidden layer is increased to two and three layers following the same procedure.
- Therefore, the structure that gives almost the same minimum average percent errors for both conditions is selected.

- The criteria for defining first hidden layer for zirconium chloride entailed test and train sample which provided the average percent error less than 60% with difference around 5%. For two hidden layers, this scale was tuned to 25% with difference below 2% and for three hidden layers; it is limited to 12% and 1.2%.
- Two final structures from result of ANI implementation on zirconium chloride which show the productive predictions are related to [9, 15, 10]-18 and [10, 11, 25]-19. Figure 5.2 compares the ANI prediction with the experimental data set of 5wt% zirconium chloride at 250 mV/s with two final structures. The RMSE of structure-1 is 0.0439 and this value for structure-2 is about 0.0451.

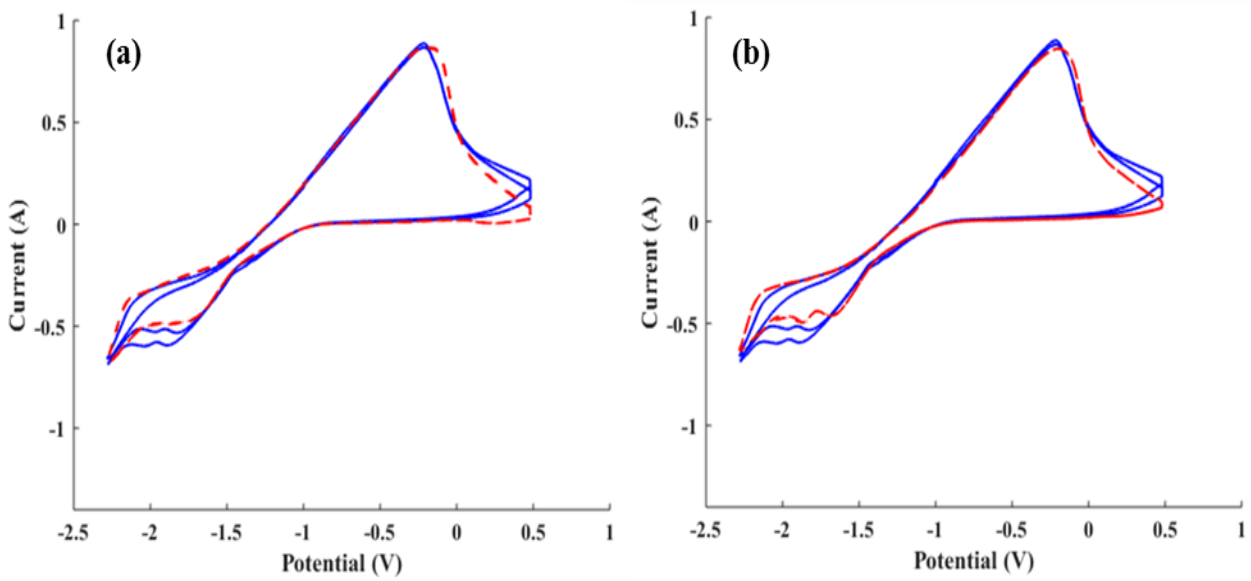

Fig. 5.2 Comparison of CV plot for 5wt% $ZrCl_4$ at 250 mV/s, (a): [9, 15, 10]-18, (b): [10, 26, 7]-20 (Blue line= experimental data, Red dash line =ANI prediction).

- These two final structures were applied on the uranium chloride data sets verifying the ANI concept.
- Three different fixed data combinations were considered (Table 4.4). The results illustrate that Table 4.4, Section (a) with [9, 15, 10]-18 provides the best prediction.
- Figure 5.3 shows the CV comparison for 5 wt% of uranium chloride at 450 mV/s with structure-1.

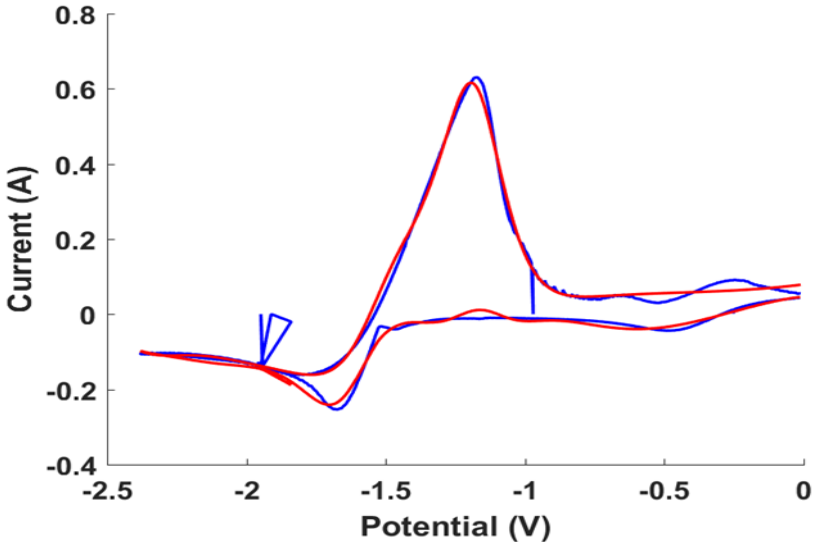

Fig. 5.3 Comparison of CV plot for 5wt% UCl_3 at 450 mV/s with structure-1.

- The diffusion model and ANI results for UCl_3 are compared in Figs. 5.4 and 5.5.

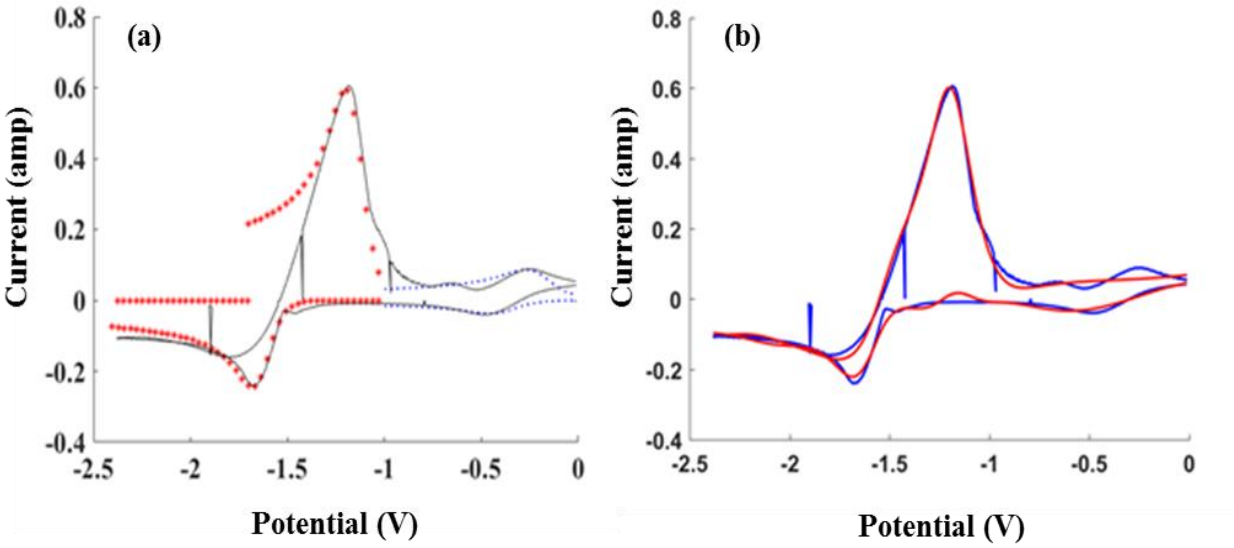

Fig. 5.4 Cyclic voltammograms of 5 wt% UCl_3 in LiCl-KCl eutectic at 773 K at the scan rate of 400 mV/s, (a): diffusion mode, (b) ANI method.

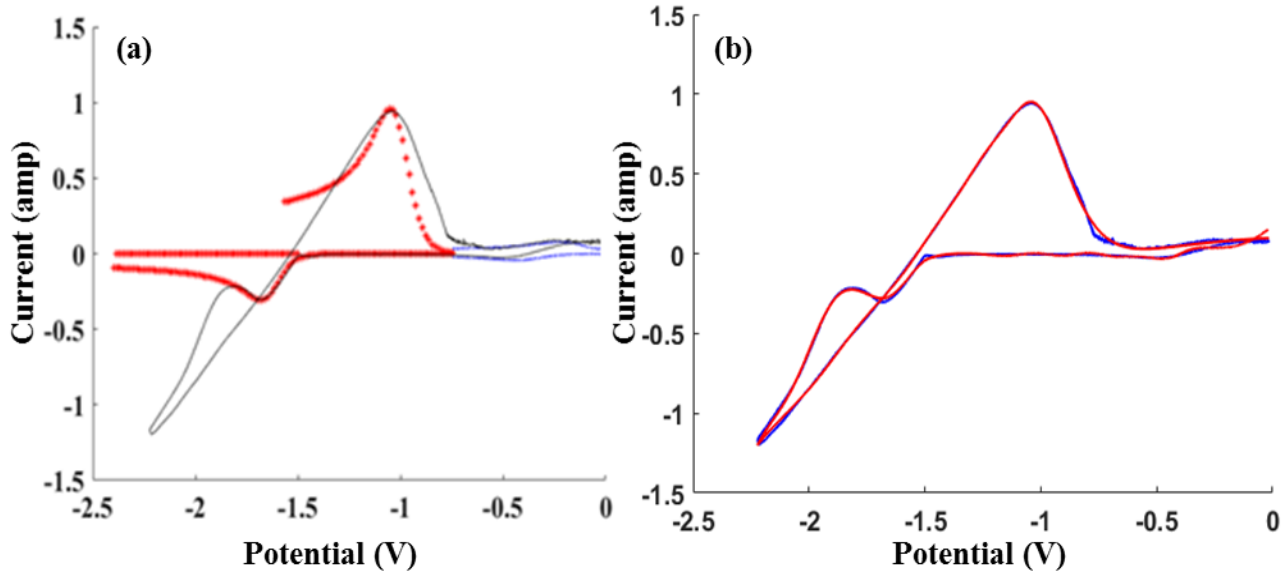

Fig. 5.5 Cyclic voltammograms of 10 wt% UCl_3 in LiCl-KCl eutectic at 773 K at the scan rate of 200 mV/s, (a): diffusion mode, (b) ANI method.

- Figs. 5.4 (b) and 5.5 (b) illustrate that the ANI is able to predict the CV without any dissimilarity and does not show any limitation in the high concentration prediction. Therefore, it is concluded that ANI can be the best method for safeguarding pyroprocessing technology due to its productivity in predicting the CV plots.

5.2 Conclusion

- There is different experimental analysis method to measure the concentration of species in electrorefinery but the sample preparation is time-consuming. Therefore, some other methods such as LIBS, UV-Vis, and CV have been proposed and explored by many researchers.
- The proposed diffusion model has been applied on uranium chloride and can capture the important charactrestic of CV method (cathodic and anodic peaks). However, the results show the limitations in terms of dissimilarity and unpredictability of irreversible side of

CV plot especially in the high concentrations. In addition, the diffusion model cannot be used for complex CV such as zirconium chloride.

- ANI doesn't have the mentioned limitation and doesn't need any technical knowlodge to implement. In addition, it can be applied on various material data sets to predict the CV and doesn't need to solve complicated equations for each material.
- Two structures ([9, 15, 10]-18, and [10, 26, 7]-20) which provide the CV with a low error have been defined and applied on the zirconium chloride data sets. The results show that [9, 15, 10]-18 structure can provide a prediction for $ZrCl_4$ data sets. It takes less than 10 minutes to train a set of experimental data with the mentioned structure and predict CVs at different concentrations and scan rates.
- It can be concluded ANI can be applied on any conditions as long as the system variables are the same (potential, current, weight percent, time, and scan rates) and the accuracy is more than 90%.
- ANI faces a limitation related to the adequate number of experimental data sets. Althought there is a need to have a huge experimental data sets, the total number of experimental data sets and training data set that provide a good prediction is not clear.
- Using a fix combination experimental data set to repeat the proposed framework from the first hidden layer is challenging; it is important to get a comprehension by applying a simple structure and improving the train and test conditions after realizing which condition is noting predicting well.
- To improve the ANI algorithem in terms of fast process time for the proposed framework, the simulation over all the sequence can be removed. This part should be used to apply

randomly on traning data set procedure to get the output in order. In addition, if the trained network will be used with new input, simulation over all segments should be considered.

5.3 Future Work

- This framework is applied on the zirconium chloride data sets and the final structure is implemented on uranium chloride experimental data sets. Although the final structure can predict the CV of uranium chloride well, it is recommended to repeat this framwork from the first step on the other existed experimental data set such as cerium chloride ($CeCl_3$), lanthanum chloride ($LaCl_3$), etc. to compare the final structure.
- Improvement on the ANI signal detection is necessary by applying on the flow sheet of pyroprocessing to understand the behavior of all elements such as U, Pu, Am, Ce, and so on.
- This study has been proposed using the number of validation checks to stop trining data sets before the process turns in overtraining. However, it may be fruitful to check if the number of epochs have any effect on overtraining.
- The levenberg-Marquardt algorithem (LMA) has been considered in this study due to its fast process time compared to other algorithem such as Bayesian Regularization algorithem (BRA). Fig. 5.6 shows for [6] structure with 5 to 25 validation numbers, the average percent error values with Bayesian Regularization is less than that of LMA. Since process time is an important factor, this is being compared in Fig. 5.7. Althought the results show that time and average error with BRA is much lower than LMA, it may happen just for this structure randomly. Further studies are necessary to complete this investigation.

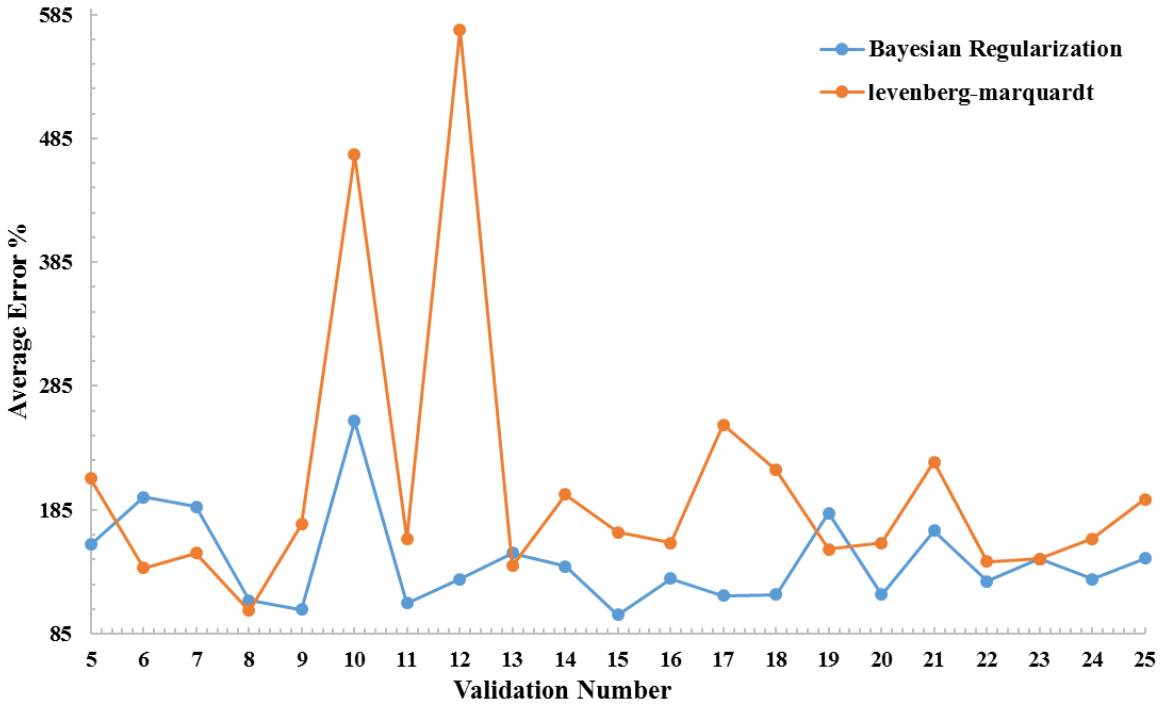

Fig. 5.6 Average error comparison for LM and BR algorithms.

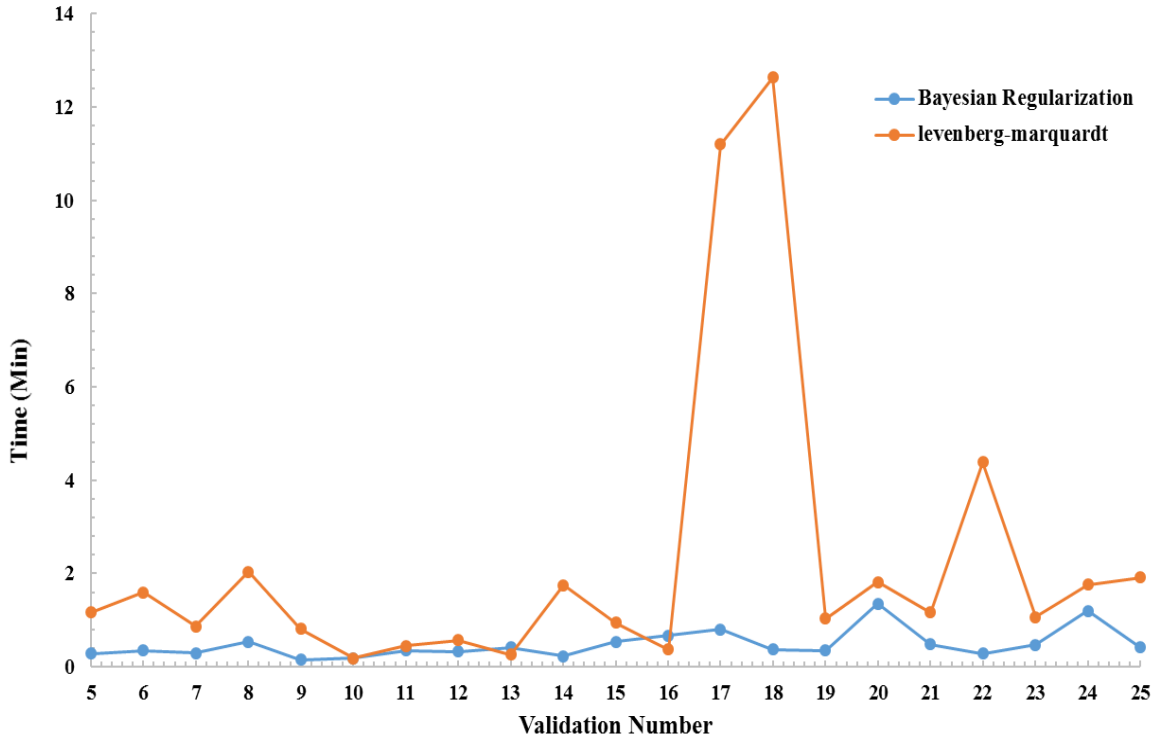

Fig. 5.7 Time comparison for LM and BR algorithms.

Chapter 6: Reference

1. Ammon N. Williams, “Measurment of Rare Earth and Uranium Elements Using Laser-induced Breakdown Spectroscopy (LIBS) in an Aerosol System for Nuclear Safegurds Applications”, Ph.D. dissertation, Mechanical and Nuclear Engineeering Department, Virginia Commonwealth University (2016).

2. P. T. Kissinger, W. R. Heineman, “Cyclic Voltammetry”, *Journal of Chemical Education*, **60** (9), pp. 702-706 (1983).

3. G. A. Mabbott, “An Introduction to Cyclic Voltammetry”, *Journal of Chemical Education*, **60** (9), pp. 697-702 (1983).

4. R. S. Nicholson, “Theory and Application of Cyclic Voltammetry for Measurement of Electrode Reaction Kinetics”, *Journal of Analytical Chemistry*, **37** (11), pp. 1351-1355 (1965).

5. Bio Logic Science Instrument, Available: http://www.bio-logic.net/en/divisions/ec-lab

6. R. O. Hoover, “Uranium and Zirconium Electrochemical Studies in LiCl-KCl Eutectic for Fundamental Applications in Used Nuclear Fuel Reprocessing", Ph.D. dissertation, Chemical Engineering Department, Idaho University (2014).

7. A. Samin, Z. Wang, E. Lahti, M. Simpson, and J. Zhang, “Estimation of Key Physical Properties for $LaCl_3$ in Molten Eutectic LiCl-KCl by Fitting Cyclic Voltammetry Data to a BET-based Electrode Reaction Kinetics Model”, *Journal of Nuclear Materials*, **475**, pp. 149-155 (2016).

8. A. Samin, E. Lahti, and J. Zhang, “Analytical Solution of the Planner Cyclic Voltammetry Process for Two Soluble Species with Equal Diffusivities and Fast Electron Transfer Using the Method of Eigenfunction Expansion”, *AIP Advances*, **5** (8), 087141 (2015).

9. R. O. Hoover, M. R. Shaltry, S. Martin, K. Sridharan, and S. Phongikaroon, "Electrochemical studies and analysis of 1–10 wt% UCl_3 concentrations in molten LiCl–KCl eutectic", *Journal of Nuclear Material*, **452** (1-3), pp. 389-396 (2014).

10. H. M. Paiva, S. F. C. Soares, and M. C. U. Araújo, "A Graphical User Interface for Variable Selection Employing the Successive Projections Algorithm", *Chemometrics and Intelligent Laboratory Systems*, 118, pp. 260-266 (2012).

11. Nuclear Technology Review 2016, Reported by Director General, GC (60)/INF/2.

12. D. Yoon, "Electrochemical Studies of Cerium and Uranium in LiCl-KCl Eutectic for Fundamentals of Pyroprocessing Technology", Ph.D. dissertation, Mechanical and Nuclear Engineering Department, Virginia Commonwealth University (2017).

13. S. Phongikaroon, EGMN 591- Nuclear Safeguards, Security, and Nonproliferation, Lecture#09, Spring (2016).

14. M. F. Simpson, "Projected Salt Waste Production from a Commercial Pyroprocesing Facility", *Science and Technology of Nuclear Installations*, 2013, Article ID 945858, http://dx.doi.org/10.1155/2013/945858.

15. Spent Nuclear Fuel Reprocessing Flowsheet, Reported by the WPFC Expert Group on Chemical Partitioning on the NEA Nuclear Science Committee, NEA/NSC/WPFC/DOC (2012)15.

16. S. Phongikaroon, EGMN 691 – Nuclear Fuel cycle, Lecture #8, Fall (2015).

17. L. J. Koch, "Experimental Breeder Reactor-II (EBR-II): An Integrated Experimental Fast Reactor Nuclear Power Station", *American Nuclear Society*, La Grange Park, Illinois, 1st Edition (2008).

18. M. F. Simpson, J. D. Law, “Nuclear Fuel Reprocessing”, Fuel Cycle Science and technology Division, Idaho National Laboratory (2010), INL/EXT-10-17753.

19. M. F. Simpson, “Developments of Spent Nuclear Fuel Pyroprocessing Technology at Idaho National Laboratory”, Prepared for U.S. Department of Energy Office of National Nuclear Security Administration Under DOE Idaho Operations Office, DE-AC07-05ID14517 (2012).

20. R. Hoover, S. Phongikaroon, S. Li, M. Simpson, and T. Yoo, “A Computational Model of Mark-IV Electrorefiner: Phase I- Fuel Basket/Salt Interface”, *Journal of Engineering for Gas Turbines and Power,* **131**(5), pp. 054503-054507 (2009).

21. S. X. Li, T. A. Johnson, B. R. Westphal, K. M. Goff, and R. W. Benedict, “Electrorefining Experience for Pyrocchemical Processing of Spent EBR-II Driver Fuel”, Proceedings of GLOBAL, Tsukuba, Japan (2005), INL/CON-05-00305.

22. “Electrometallurgical Techniques for DOE spend Fuel Treatment”, National Research Council, National Academy Press, Washington DC (2000).

23. J. Oark, Sungyeol Choi, Sungjune Sohn, Kwang-Rang Kim, and II Soon Hwang, “Cyclic Voltammetry on Zirconium Redox Reactions in LiCl-KCl-$ZrCl_4$ at 500°C for Electrorefining Contaminated Zircaloy-4 Cladding”, **161** (3), pp. H97-H1054 (2014).

24. Educator's Reference Guide for Electrochemistry, Pine Instrument Company, Grove City, Pennsylvania (2000).

25. D. Andrienko, “Cyclic Voltammetry”, John Wiely & Sons pub., New York, pp. 3-12 (2008).

26. A. J. Bard, and L. R. Faulkner, “Electrochemical Methods: Fundamentals and Applications”, Second Edition, *Wiley*, 2000.

27. S. A. Kuznetsov, H. Hayashi, K. Minato, and M. Gaune-Escard, "Electrochemical Behavior and Some Thermodynamic Properties of UCl_4 and UCl_3 Dissolved in a LiCl-KCl Eutectic Melt" *Journal of the Electrochemical Society*, **152**(4), C203-C212 (2005).

28. P. Masset, D. Bottomley, R. Konings, R. Malmbeck, A. Rodrigues, J. Serp, and J. Glatz, "Electrochemistry of Uranium in Molten LiCl-KCl Eutectic", *Journal Electrochemical Society*, **152** (6), pp. A1109-A1115 (2005).

29. P. Delahay, "Theory of Irreversible Waves in Oscillographic Polarography", *Journal of the American Chemical Society*, **75** (5), pp. 1190-1195 (1953).

30. "Density of Molten Elements and Representative Salts", *CRC Handbook of Chemistry and Physics*, Chapter 4, pp. 126-129 (2003).

31. G. J. Janz and N. P. Bansal, "Molten Salts Data: Diffusion Coefficients in Single and Multi-Component Salt Systems," Journal of Physical and Chemical Reference Data, 11(3), 505-693 (1982).

32. T. Kobayashi, R. Fujita, M. Fujie, and T. Koyama, "Polarization Effects in the Molten Salt Electrorefining of Spent Nuclear Fuel," *Journal of Nuclear Science and Technology*, **32** (7), pp. 53-63 (1995).

33. C. E. Thalmayer, S. Bruckenstien, and D. M. Gruen, "Chronopotentiometric Determination of Interdiffusion Coefficients and Heats of Interdiffusion in Molten Salts", *Journal of Inorganic and Nuclear Chemistry*, **26** (2), pp. 347- 357 (1964).

34. R. S. Nicholson and I. Shain, "Theory of Stationary Electrode Polarography Single Scan and Cyclic Methods Applied to Reversible, Irreversible, and Kinetic System", *Journal of Analytical Chemistry*, **36** (4), pp. 706-723 (1964).

35. D. Inman, G. J. Hills, L. Young, and J. O'M Bockris, "Electrode Reactions in Molten Salts: The Uranium + Uranium Trichloride System", *Transactions of the Faraday Society,* **55**, pp. 1904–1914 (1959).

36. D. L. Hill, J. Perano, and R. A. Osteryoung, "An Electrochemical Study of Uranium in Fused Chlorides", *Journal of Electrochemical Society,* **107** (8), pp. 698-705 (1960).

37. D. Inman, and J. O'M. Bockris, "The Reversible Electrode Potential of the System U/UCl_3 in Molten Chloride Solvents", *Canadian Journal of Chemistry,* **39** (5), pp. 1161-1163 (1961).

38. J. J. Roy, L. F. Grantham, D. L. Grimmett, S. P. Fusselman, C. L. Krueger, T. S. Storvick, T. Inoue, Y. Sakamura, and N. Takahashi, "Thermodynamic Properties of U, Np, Pu, and Am in Molten LiCl-KCl Eutectic and Liquid Cadmium", *Journal of Electrochemical Society,* **143** (8), pp. 2487-2492 (1996).

39. Y. Sakamura, T. Hijikata, K. Kinoshita, T. Inoue, T. S. Storvick, C. L. Krueger, Jay Jyoti Roy, D. L. Grimmett, S. P. Fusselman, R. L. Gay, "Measurement of Standard Potentials of Actinides (U, Np, Pu, Am) in LiCl-KCl Eutectic Salt and Separation of Actinides from Rare Earths by Electrorefining", *Journal of Alloys and Compounds,* **271-273**, pp. 592-596 (1995).

40. I. Choi, B. E. Serrano, S. X. Li, S. Herman, S. Phongikaroon, " Determination of Exchange Current Density of U^{+3}/U Couple in LiCl-KCl Eutectic Mixture", *Proceeding of International Conference Global*, September 6-11, Paris, France (2009).

41. O. Shirai, H. Yamana, Y. Arai, "Electrochemical Behavior of Actinides and Actinide Nitrides in LiCl-KCl Eutectic Melts", *Journal of Alloys and Compounds,* **408-412**, pp. 1267-1273 (2006).

42. S. H. Kim, D. S. Yoon, Y. J. You, S. Paek, J. B. Shim, S. W. Kwon, K. R. Kim, H. S. Chung, D. H. Ahn, and H. S. Lee, "In-situ Observation of a Dendrite Growth in an Aqueous

Condition and a Uranium Deposition into a Liquid Cadmium Cathode in an Electrowinning System", *Journal of Nuclear Materials,* **385** (1), pp. 196-199 (2009).

43. W. H. Reinmuth, "Theory of Diffusion Limited Charge-Transfer Processes in Electroanalytical Techniques", *Journal of Analytical Chemistry*, **34**, pp. 1446-1454 (1962).

44. W. H. Reinmuth, "Theory of stationary Electrode Polarography", *Journal of Analytical Chemistry,* **33** (12), pp. 1793-1794 (1961).

45. K. B. Oldham, "Analytical Expressions for the Reversible Randles-Sevcik Function", *Journal of Electroanalytical Chemical*, **105**, pp. 373-375 (1979).

46. Y. Cai, H.a Liu, Q. Xu, Q. Song, H. Liu, and L. Xu, "Electrochemical Behavior of Zirconium in an in-situ Preparing LiCl-KCl-$ZrCl_4$ Molten Salt", *International Journal of Electrochemical Science*, **10**, pp. 4324-4334 (2015).

47. Y. Sakamura, "Zirconium Behavior in Molten LiCl-KCl Eutectic", *Journal of the Electrochemical Society*, 151 (3), pp. C187-C193 (2004).

48. C. P. Fabian, V. Luca, T. H. Le, A. M. Bond, P. Chamelot, L. Massot, C. Caravaca, T. L. Hanley, and G. R. Lumpkin, "Cyclic Voltammetric Experiment-Simulation Comparisons of the Complex Mechanism Associated with Electrochemical Reduction of Zr^{+4} in LiCl-KCl Eutectic Molten Salt", *Journal of the Electrochemical Society*, **160** (2), pp. H81-H86 (2013).

49. S. Lahiri, and K. C. Ghanta, "Artificial Neural Network Model with Parameter Tuning Assisted by Agentic Algorithm Technique: Study of Critical Velocity of Slurry Flow in Pipeline", *Asia-Pacific Journal of Chemical Engineering,* **15** (2), pp.763-777 (2010).

50. D. Kriesel, " A Brief Introduction to Neural Networks", 2007, Available at: http://www.dkriesel.com/_media/science/neuronalenetze-en-zeta2-2col-dkrieselcom.pdf.

51. D. Wijayasekara, M. Manic, P. Sabharwall, and V. Utgikar, “Optimal Artificial Neural Network Architecture Selection for Performance Prediction of Compact Heat Exchanger with the EBaLM-OTR Technique”, *Nuclear Engineering and Design*, **241** (7), pp.2549-2557 (2011).

52. A. Ridluan, M. Manic, and A. Tokuhiro, “EBaLM-THP-a Neural Network Thermo hydraulic Prediction model of Advanced Nuclear System Components”, *Nuclear Engineering and Design*, **239** (2), pp.308-319 (2009).

53. M. Manic, P. Sabharwall, "Computational Intelligence as a Tool for Small Modular Reactors," *Small Modular Reactors Symposium (SMR)*, ASME 2011 SMR, Washington DC, September, pp.28-30 (2011).

54. M. A. Nielson, “Neural Networks and Deep Learning”, Determination Press, 2015. Available: http://neuralnetworksanddeeplearning.com/index.html

55. D. O. Araromi, T. J. Afolabi, “Neural Network Control of CSTR for Reversible Reaction Using Reverence Model Approach”, *Leonardo Journal of Sciences*, **6**(10), pp.25-40 (2007).

56. A. S. Planche, N. D. S. Cordeiro, “A General ANN-Based Multitasking Model for the Discovery of Potent and Safer Antibacterial Agents”, *Methods in Molecular Biology*, **1260**, pp.45-64 (2015).

57. P. Siriphala, “Controlling Artificial Neural Networks Overtraining When Data Is Scarce”, Ph.D. Dissertation, Wichita State University, Department of Industrial and Manufacturing Engineering (2000).

58. S. Lawrence, C. L. Giles, and A. C. Tsoi,” Lessons in Neural Network Training: Overfitting May Be Harder than Expected”, *Proceedings of the Fourteenth National Conference on Artificial Intelligence*, AAAI-97, AAAI Press, pp.540–545, 1997, Menlo Park, California.

59. I. I. Baskin, V. A. Palyulin, and N. S. Zefirov, “Neural Networks in Building QSAR Models”, Artificial Neural Networks, Series: Methods in Molecular Biology, Humana Press, **458**, pp.133-154 (2008).

60. F. Burden, and D. Winkler, “Bayesian Regularization of Neural Networks”, *Method in Molecular Biology* ^{TM}, 458, pp.23-42 (2009).

61. B. Taylor, M. Darrah, and C. Moats, “Verification and Validation of Neural Networks: a Sampling of Research in Progress”, *Proceeding in the International Society for Optical Engineering (SPIE)*, Intelligent Computing: Theory and Applications, **5103** (8), August 7 (2003).

62. H. P. Gavin, “The Levenberg-Marquardt Method for Nonlinear Least Squares Curve-Fitting Problems”, Duke University (2016), Available at http://people.duke.edu/~hpgavin/ce281/lm.pdf

63. S. Rakhshan Pouri, S. Phongikaroon, “Investigation on Reactions Probabilities for Cyclic Voltammetry of Zirconium in LiCl-KCl Eutectic Molten Salt via Reverse-Engineering Method”, American Nuclear Society (ANS), Annual meeting, New Orleans, LA, June 12-16, 2016.

Appendix I: *Matlab* Code for Diffusion Coefficient and Apparent Standard Potential

I.1 Uranium Chloride

```
    clc
    clear

    [status,sheets] = xlsfinfo('rawdata-U-1.xlsx')
    S=input(' Start Sheet Number (Please Put 2 if you are considering whole
Sheets) = ');
    L=input(' End Sheet Number in Excel File = ');
    Equation=input(' Put 1 for Reversible and 2 For Irreversible = ');
    n=input(' Number of Electron Transffered = ');

    T=773.15;  % K
    R=8.314;   % J/mol.K
    F=96485;   % C/eq
    alpha=0.5;
    k=0.00026; % cm/s

    for ii=S:L
    sheetii=xlsread('rawdata-U-1.xlsx',ii);
    sheet1=xlsread('rawdata-U-1.xlsx',1);
    plot(sheetii(:,1),sheetii(:,2));
    hold all
    [x,y]=ginput(1);
    Ipc(ii)=y;
    Epc(ii)=x;
    Ipeak=Ipc(1,ii);
    Epeak=Epc(1,ii);
    W=sheet1(ii-1,1);
    ScanRate=sheet1(ii-1,2);
    if W==1;
        A=0.626;
    elseif W==2.5;
        A=0.583;
    elseif W==5;
         A=0.710;
    elseif W==7.5;
         A=0.659;
    else W==10;
        A=0.785;
    end

    M=344.39 ; % UCl3
    RumLiCl=1.502; % g/cm3
```

```
    RumKCl=1.527;
    tmLiCl=610; % C
    tmKCl=771; % C
    kLiCl=0.000432; % g/cm3C
    kKCl=0.000583; % g/cm3C
    RuLiCl=RumLiCl-(kLiCl*((T-274.14)-tmLiCl));
    RuKCl=RumKCl-(kKCl*((T-274.14)-tmKCl));
    Ru=(0.5*RuKCl)+(0.5*RuLiCl);
    C=(W/100)*Ru/M; % mol/cm3

    % 2= Irreversible 1=Reversible

    if Equation==2;

Dwithalpha2(ii)=((Ipeak^2)*R*T)/(((0.496*n*F*A*C)^2)*(ScanRate/1000)*F*n*alph
a);
    DNoalpha2(ii)=((Ipeak^2)*R*T)/(((0.496*n*F*A*C)^2)*(ScanRate/1000)*F*n);
    E0withalpha2(ii)=Epeak+((R*T/(n*alpha*F))*(0.78-
log(k)+log(sqrt((n*alpha*F*(ScanRate/1000)*Dwithalpha2(1,ii))/(R*T)))));
    E0withnoalpha2(ii)=Epeak+((R*T/(n*alpha*F))*(0.78-
log(k)+log(sqrt((n*alpha*F*(ScanRate/1000)*DNoalpha2(1,ii))/(R*T)))));
    else Equation==1;

Dwithalpha1(ii)=((Ipeak^2)*R*T)/(((0.446*n*F*A*C)^2)*(ScanRate/1000)*F*n*alph
a);
    DNoalpha1(ii)=((Ipeak^2)*R*T)/(((0.446*n*F*A*C)^2)*(ScanRate/1000)*F*n);
    E0withalpha1(ii)=Epeak+((R*T/(n*alpha*F))*(0.78-
log(k)+log(sqrt((n*alpha*F*(ScanRate/1000)*Dwithalpha1(1,ii))/(R*T)))));
    E0withnoalpha1(ii)=Epeak+((R*T/(n*alpha*F))*(0.78-
log(k)+log(sqrt((n*alpha*F*(ScanRate/1000)*DNoalpha1(1,ii))/(R*T)))));
    end
    end

    IpeakAverage=sum(Ipc)/(L-1)
    STDEVIpeak=std(Ipc(:,S:L))
    EpeakAverage=sum(Epc)/(L-1)
    STDEVEpeak=std(Epc(:,S:L))

    if Equation==2;

    DwithalphaIrreversible=sum(Dwithalpha2)/(L-1)
    STDEVDwithalpha=std(Dwithalpha2(:,S:L))
    DNoalphaIrreversible=sum(DNoalpha2)/L
    STDEVDNoalpha=std(DNoalpha2(:,S:L))
    EwithalphaIrreversible=sum(E0withalpha2)/(L-1)
    STDEVEwithalpha=std(E0withalpha2(:,S:L))
    ENoalphaIrreversible=sum(E0withnoalpha2)/(L-1)
    STDEVENoalpha=std(E0withnoalpha2(:,S:L))

    else Equation==1;
    DwithalphaIrreversible=sum(Dwithalpha1)/(L-1)
    STDEVDwithalpha=std(Dwithalpha1(:,S:L))
    DNoalphaIrreversible=sum(DNoalpha1)/L
    STDEVDNoalpha=std(DNoalpha1(:,S:L))
    EwithalphaIrreversible=sum(E0withalpha1)/(L-1)
```

```
    STDEVEwithalpha=std(E0withalpha1(:,S:L))
    ENoalphaIrreversible=sum(E0withnoalpha1)/(L-1)
    STDEVENoalpha=std(E0withnoalpha1(:,S:L))
    end
```

I.2 Zirconium Chloride

```
    clc
    clear
    [status,sheets] = xlsfinfo('rawdata-zr.xlsx')
    S=input(' Start Sheet Number (Please Put 2 if you are considering whole
Sheets) = ');
    L=input(' End Sheet Number in Excel File = ');
    n=input(' Number of Electron Transffered = ');

    T=773.15; % K
    R=8.314;  % J/mol.K
    F=96485;  % C/eq
    alpha=0.5;
    k=0.00026;% cm/s

    for ii=S:L;
    sheetii=xlsread('rawdata-zr.xlsx',ii);
    sheet1=xlsread('rawdata-zr.xlsx',1);
    plot(sheetii(:,1),sheetii(:,2));
    hold all
    [x,y]=ginput(1);
    Ipc(ii)=y;
    Epc(ii)=x;
    Ipeak=Ipc(1,ii);
    Epeak=Epc(1,ii);
    W=sheet1(ii-1,1);
    ScanRate=sheet1(ii-1,2);
    if W==0.57;
        A=2.325;
    elseif W==1.07;
        A=2.325;
    elseif W==2.49;
         A=2.450;
    else W==4.98;
         A=2.545;
    end

    M=233.04; % Zrcl4
    RumLiCl=1.502; % g/cm3
    RumKCl=1.527;
    tmLiCl=610; % C
    tmKCl=771; % C
    kLiCl=0.000432; % g/cm3C
    kKCl=0.000583; % g/cm3C
    RuLiCl=RumLiCl-(kLiCl*((T-274.14)-tmLiCl));
    RuKCl=RumKCl-(kKCl*((T-274.14)-tmKCl));
    Ru=(0.5*RuKCl)+(0.5*RuLiCl);
```

```
    C=(W/100)*Ru/M; % mol/cm3

    % 2= Irreversible


Dwithalpha2(ii)=((Ipeak^2)*R*T)/(((0.496*n*F*A*C)^2)*(ScanRate/1000)*F*n*alph
a);
    DNoalpha2(ii)=((Ipeak^2)*R*T)/(((0.496*n*F*A*C)^2)*(ScanRate/1000)*F*n);
    E0withalpha2(ii)=Epeak+((R*T/(n*alpha*F))*(0.78-
log(k)+log(sqrt((n*alpha*F*(ScanRate/1000)*Dwithalpha2(1,ii))/(R*T)))));
    E0withnoalpha2(ii)=Epeak+((R*T/(n*alpha*F))*(0.78-
log(k)+log(sqrt((n*alpha*F*(ScanRate/1000)*DNoalpha2(1,ii))/(R*T)))));
    end


    IpeakAverage=sum(Ipc)/(L-1)
    STDEVIpeak=std(Ipc(:,S:L))
    EpeakAverage=sum(Epc)/(L-1)
    STDEVEpeak=std(Epc(:,S:L))
    DwithalphaIrreversible=sum(Dwithalpha2)/(L-1)
    STDEVDwithalpha=std(Dwithalpha2(:,S:L))
    DNoalphaIrreversible=sum(DNoalpha2)/L
    STDEVDNoalpha=std(DNoalpha2(:,S:L))
    EwithalphaIrreversible=sum(E0withalpha2)/(L-1)
    STDEVEwithalpha=std(E0withalpha2(:,S:L))
    ENoalphaIrreversible=sum(E0withnoalpha2)/(L-1)
    STDEVENoalpha=std(E0withnoalpha2(:,S:L))
```

Appendix II: Molten Salt Density (Ref.30)

Table II.1 Density of molten elements and representative salts (Ref. 30).

Formula	Name	t_m/ C	ρ_m/g cm^{-3}	k/g cm^{-3} °C^{-1}	t_{max}
Ag	Silver	961.78	9.320	0.0009	1500
AgBr	Silver(I) bromide	432	5.577	0.001035	667
AgCl	Silver(I) chloride	455	4.83	0.00094	627
AgI	Silver(I) iodide	558	5.58	0.00101	802
$AgNO_3$	Silver(I) nitrate	212	3.970	0.001098	360
Ag_2SO_4	Silver(I) sulfate	652	4.84	0.001089	770
Al	Aluminum	660.32	2.375	0.000233	1340
$AlBr_3$	Aluminum bromide	97.5	2.647	0.002435	267
$AlCl_3$	Aluminum chloride	192.6	1.302	0.002711	296
AlI_3	Aluminum iodide	188.32	3.223	0.0025	240
As	Arsenic	817	5.22	0.000544	
Au	Gold	1064.18	17.31	0.001343	1200
B	Boron	2075	2.08		
Ba	Barium	727	3.338	0.000299	1550
$BaBr_2$	Barium bromide	857	3.991	0.000924	900
$BaCl_2$	Barium chloride	962	3.174	0.000681	1081
BaF_2	Barium fluoride	1368	4.14	0.000999	1727
BaI_2	Barium iodide	711	4.26	0.000977	975
Be	Beryllium	1287	1.690	0.00011	
$BeCl_2$	Beryllium chloride	415	1.54	0.0011	473
BeF_2	Beryllium fluoride	552	1.96	0.000015	850
Bi	Bismuth	271.40	10.05	0.00135	800
$BiBr_3$	Bismuth bromide	218	4.76	0.002637	927
$BiCl_3$	Bismuth chloride	230	3.916	0.0023	350
Ca	Calcium	842	1.378	0.000230	1484
$CaBr_2$	Calcium bromide	742	3.111	0.0005	791
$CaCl_2$	Calcium chloride	775	2.085	0.000422	950
CaF_2	Calcium fluoride	1418	2.52	0.000391	2027
CaI_2	Calcium iodide	783	3.443	0.000751	1028
Cd	Cadmium	321.07	7.996	0.001218	500

$CdBr_2$	Cadmium bromide	568	4.075	0.00108	720
$CdCl_2$	Cadmium chloride	564	3.392	0.00082	807
CdI_2	Cadmium iodide	387	4.396	0.001117	700
Ce	Cerium	799	6.55	0.000710	1460
$CeCl_3$	Cerium(III) chloride	817	3.25	0.00092	950
CeF_3	Cerium(III) fluoride	1430	4.659	0.000936	1927
Co	Cobalt	1495	7.75	0.00165	1580
Cr	Chromium	1907	6.3	0.0011	2100
Cs	Cesium	28.44	1.843	0.000556	510
CsBr	Cesium bromide	636	3.133	0.001223	860
CsCl	Cesium chloride	645	2.79	0.001065	906
CsF	Cesium fluoride	703	3.649	0.001282	912
CsI	Cesium iodide	621	3.197	0.001183	907
$CsNO_3$	Cesium nitrate	414	2.820	0.001166	491
Cs_2SO_4	Cesium sulfate	1005	3.1	0.00095	1530
Cu	Copper	1084.62	8.02	0.000609	1630
CuCl	Copper(I) chloride	430	3.692	0.00076	585
Dy	Dysprosium	1411	8.37	0.00143	1540
$DyCl_3$	Dysprosium(III) chloride	680	3.62	0.00068	987
Er	Erbium	1529	8.86	0.00157	1700
Eu	Europium	822	5.13	0.0028	980
Fe	Iron	1538	6.98	0.000572	1680
$FeCl_2$	Iron(II) chloride	677	2.348	0.000555	877
Ga	Gallium	29.76	6.08	0.00062	400
$GaBr_3$	Gallium(III) bromide	121.5	3.116	0.00246	135
$GaCl_3$	Gallium(III) chloride	77.9	2.053	0.002083	141
GaI_3	Gallium(III) iodide	212	3.630	0.002377	252
Gd	Gadolinium	1314	7.4		
$GdCl_3$	Gadolinium(III) chloride	609	3.56	0.000671	1007
GdI_3	Gadolinium(III) iodide	925	4.12	0.000908	1032
Ge	Germanium	938.25	5.60	0.00055	1600
Hf	Hafnium	2233	12		
$HgBr_2$	Mercury(II) bromide	236	5.126	0.003233	319
$HgCl_2$	Mercury(II) chloride	276	4.368	0.002862	304
HgI_2	Mercury(II) iodide	259	5.222	0.003235	354
Ho	Holmium	1472	8.34		
In	Indium	156.60	7.02	0.000836	500
$InBr_3$	Indium(III) bromide	420	3.121	0.0015	528

$InCl_3$	Indium(III) chloride	583	2.140	0.0021	666
InI_3	Indium(III) iodide	207	3.820	0.0015	360
Ir	Iridium	2446	19		
K	Potassium	63.38	0.828	0.000232	500
KBr	Potassium bromide	734	2.127	0.000825	930
KCl	Potassium chloride	771	1.527	0.000583	939
KF	Potassium fluoride	858	1.910	0.000651	1037
KI	Potassium iodide	681	2.448	0.000956	904
KNO_3	Potassium nitrate	337	1.865	0.000723	457
La	Lanthanum	920	5.94	0.00061	1600
$LaBr_3$	Lanthanum bromide	788	4.933	0.000096	912
$LaCl_3$	Lanthanum chloride	859	3.209	0.000777	973
LaF_3	Lanthanum fluoride	1493	4.589	0.000682	2177
LaI_3	Lanthanum iodide	778	4.29	0.001110	907
Li	Lithium	180.5	0.512	0.00052	285
LiBr	Lithium bromide	552	2.528	0.000652	739
LiCl	Lithium chloride	610	1.502	0.000432	781
LiF	Lithium fluoride	848.2	1.81	0.000490	1047
LiI	Lithium iodide	469	3.109	0.000917	667
$LiNO_3$	Lithium nitrate	253	1.781	0.000546	441
Li_2SO_4	Lithium sulfate	859	2.003	0.000407	1214
Lu	Lutetium	1663	9.3		
Mg	Magnesium	650	1.584	0.000234	900
$MgBr_2$	Magnesium bromide	711	2.62	0.000478	935
$MgCl_2$	Magnesium chloride	714	1.68	0.000271	826
MgI_2	Magnesium iodide	634	3.05	0.000651	888
Mn	Manganese	1246	5.95	0.00105	1590
$MnCl_2$	Manganese(II) chloride	650	2.353	0.000437	850
Mo	Molybdenum	2623	9.33		
Na	Sodium	97.80	0.927	0.00023	600
NaBr	Sodium bromide	747	2.342	0.000816	945
Na_2CO_3	Sodium carbonate	858.1	1.972	0.000448	1004
NaCl	Sodium chloride	800.7	1.556	0.000543	1027
NaF	Sodium fluoride	996	1.948	0.000636	1097
NaI	Sodium iodide	660	2.742	0.000949	912
$NaNO_3$	Sodium nitrate	307	1.90	0.000715	370
Na_2SO_4	Sodium sulfate	884	2.069	0.000483	1077
Nd	Neodymium	1016	6.89	0.00076	1350
Ni	Nickel	1455	7.81	0.000726	1700
$NiCl_2$	Nickel(II) chloride	1009	2.653	0.00066	1057

Os	Osmium	3033	20		
Pb	Lead	327.46	10.66	0.00122	700
$PbBr_2$	Lead(II) bromide	371	5.73	0.00165	600
$PbCl_2$	Lead(II) chloride	501	4.951	0.0015	710
PbI_2	Lead(II) iodide	410	5.691	0.001594	697
Pd	Palladium	1554.9	10.38	0.001169	1700
Pr	Praseodymium	931	6.50	0.00093	1460
$PrCl_3$	Praseodymium chloride	786	3.23	0.00074	977
Pt	Platinum	1768.4	19.77	0.0024	2200
Pu	Plutonium	640	16.63	0.001419	950
Rb	Rubidium	39.31	1.46	0.000451	800
RbBr	Rubidium bromide	682	2.715	0.001072	907
Rb_2CO_3	Rubidium carbonate	837	2.84	0.000640	1007
RbCl	Rubidium chloride	715	2.248	0.000883	923
RbF	Rubidium fluoride	833	2.87	0.00102	1067
RbI	Rubidium iodide	642	2.904	0.001143	902
$RbNO_3$	Rubidium nitrate	305	2.519	0.001068	417
Rb_2SO_4	Rubidium sulfate	1050	2.56	0.000665	1545
Re	Rhenium	3186	18.9		
Rh	Rhodium	1964	10.7	0.000895	2200
Ru	Ruthenium	2334	10.65		
S	Sulfur	115.21	1.819	0.00080	160
Sb	Antimony	630.63	6.53	0.00067	745
$SbCl_5$	Antimony(III) chloride	73.4	2.681	0.002293	77
$SbCl_5$	Antimony(V) chloride	4	2.37	0.001869	77
SbI_3	Antimony(III) iodide	168	4.171	0.002483	322
Sc	Scandium	1541	2.80		
Se	Selenium	221	3.99		
Si	Silicon	1414	2.57	0.000936	1500
Sm	Samarium	1072	7.16		
Sn	Tin	231.93	6.99	0.000601	1200
$SnCl_2$	Tin(II) chloride	247	3.36	0.001253	480
$SnCl_4$	Tin(IV) chloride	-33	2.37	0.002687	138
Sr	Strontium	777	6.980		
$SrBr_2$	Strontium bromide	657	3.70	0.000745	1004
$SrCl_2$	Strontium chloride	874	2.727	0.000578	1037
SrF_2	Strontium fluoride	1477	3.470	0.000751	1927
SrI_2	Strontium iodide	538	4.085	0.000885	1026
Ta	Tantalum	3017	15		
$TaCl_5$	Tantalum(V) chloride	216	2.700	0.004316	457

Tb	Terbium	1359	7.65		
Te	Tellurium	449.51	5.70	0.00035	600
$ThCl_4$	Thorium chloride	770	3.363	0.0014	847
ThF_4	Thorium fluoride	1110	6.058	0.000759	1378
Ti	Titanium	1668	4.11		
$TiCl_4$	Titanium(IV) chloride	-25	1.807	0.001735	137
Tl	Thallium	304	11.22	0.00144	600
TlBr	Thallium(I) bromide	460	5.98	0.001755	647
TlCl	Thallium(I) chloride	430	5.628	0.0018	642
TlI	Thallium(I) iodide	441.8	6.15	0.001761	737
$TlNO_3$	Thallium(I) nitrate	206	4.91	0.001873	279
Tl_2SO_4	Thallium(I) sulfate	632	5.62	0.00130	927
Tm	Thulium	1545	8.56	0.00050	1675
U	Uranium	1135	17.3		
UCl_3	Uranium(III) chloride	837	4.84	0.007943	1057
UCl_4	Uranium(IV) chloride	590	3.572	0.001945	667
UF_4	Uranium(IV) fluoride	1036	6.485	0.000992	1341
V	Vanadium	1910	5.5		
W	Tungsten	3422	17.6		
Y	Yttrium	1526	4.24		
YCl_3	Yttrium chloride	721	2.510	0.0005	845
Yb	Ytterbium	824	6.21		
Zn	Zinc	419.53	6.57	0.0011	700
$ZnBr_2$	Zinc bromide	394	3.47	0.000959	602
$ZnCl_2$	Zinc chloride	290	2.54	0.00053	557
ZnI_2	Zinc iodide	446	3.878	0.00136	588
$ZnSO_4$	Zinc sulfate	680	3.14	0.00047	987
Zr	Zirconium	1855	5.8		
$ZrCl_4$	Zirconium chloride	437	1.643	0.007464	492

Appendix III: Thermodynamic Values for UCl_3

Summary of diffusion coefficients and apparent standard potentials values for UCl_3 reported in the literatures are listed in Table III.1. The main part of this report is based on Hoover's study (Ref. 6). In addition, results of the cathodic peak current versus square root of scan rate are shown in this Appendix.

Table III.1 Summary of diffusion coefficient and apparent standard potential of U^{+4}/U^{+3} and U^{+3}/U^{+4} reported in literatures (Ref. 6).

References	$D(U^{+3}/U)$ ($\times 10^5$ cm^2/s)	$D(U^{+4}/U^{+3})$ ($\times 10^5 cm^2/$)	$E^{o'}$ (U^{+3}/U) (V vs Cl_2/Cl^-)	$E^{o'}$ (U^{+4}/U^{+3}) (V vs Cl_2/Cl^-)		T (°C)
Boussier[3] (2003)	1.21		-2.511		Cl_2/Cl^-	500
Bychkov et al.(2000)			-2.31	-1.4	Cl_2/Cl^-	500
Caligara et al. (1967)	1.031	1.215				500
Caligara et al.(Pt) (1967)	0.68	0.8				450
Caligara et al. (1967) (Pt-Correlation)	0.9337	0.9638				500
Caligara et al. (1967) (Correlation)	0.7105	1.027				500
Choi et al. (2009)	1.06					500
Gao Fanxing et al.(2009)	8.1		-2.47		Cl_2/Cl^-	455
Gao Fanxing et al.(2009)			-1.40		Ag/AgCl	455
Gruen et al. (1960)			-2.54		Cl_2/Cl^-	450
Hill et al. (1960)			-2.47	-1.466	Cl_2/Cl^-	450

3 - Electrochemistry of Uranium: The Solution of Uranium was prepared by UO_2

Hill et al. (1960)				-1.25	Pt(II)/Pt	450
Hill et al. (1960)			-1.61		Ag(I)/Ag	450
Hill et al. (1960)			-2.25		Pt(II)/Pt	450
Hoover (CV) (2014)	2.52	0.126	-1.502	-0.381	Ag/AgCl	500
Hoover (CP) (2014)	1.04	0.672	-1.502	-0.381	Ag/AgCl	500
Inman et al.(1959)	6					450
Inman and Bockris (1961)			-2.671		Cl_2/Cl^-	453
Kim et al.(2009)	1				Cl_2/Cl^-	500
Kim et al.(2009)	1.03		-1.24		Ag/AgCl	
Kuznetsov et al.(Pt) (2005)	1.02	0.68	-2.541		Cl_2/Cl^-	450
Kuznetsov et al. (2005)	1.45		-2.514		Cl_2/Cl^-	500
Martinot et al. (1970)		1.22	-2.489	-1.469	Cl_2/Cl^-	500
Martinot et al. (1970)				-1.492	Cl_2/Cl^-	450
Martinot and Caligara (1973)			-2.527		Cl_2/Cl^-	450
Martinot and Caligara (1973)			-2.491		Cl_2/Cl^-	500
Martinot and Caligara (1973)			-2.483		Cl_2/Cl^-	500
Martinot and Caligara (1973)				-1.521	Cl_2/Cl^-	450
Martinot and Caligara (1973)				-1.485	Cl_2/Cl^-	500
Masset et al. (2005)	2.5		-2.516	-1.415	Cl_2/Cl^-	500

Masset et al. (2005)			-1.281	-0.181	Ag/AgCl	500
Masset et al. (2005) (Correlation)	3.175	2.4122	-2.504	-1.43	Cl_2/Cl^-	500
Masset et al. (2005)	3.8		-2.491		Cl_2/ Cl^-	500
Masset et al. (CP) (2005)			-1.257		Ag/AgCl	500
Masset et al. (2005)	1.36	0.27	-2.571	-1.479	Cl_2/Cl^-	430
Masset et al.(2005)			-1.281		Ag/AgCl	430
Reddy et al. (2004)	0.98					497
Reddy et al. (2004)			-1.490	-0.325	Ag/AgCl	387-505
Roy et al.(1996)			-2.498			450
Sakamura et al. (1998)			-1.283		Ag/AgCl	450
Shirai et al.(1998)			-2.4533	-1.487	Cl_2/Cl^-	500
Shirai et al.(1998)			-1.2478		Ag/AgCl	500
Shirai et al.(1998)			-2.484	-1.495	Cl_2/Cl^-	450
Shirai et al. (1998) (Correlation)				-1.4959	Cl_2/Cl^-	500
Zhang (2014)	1.019	0.6826	-2.5183	-1.5171	Cl_2/Cl^-	450-550
Thalmayer et al. (1964)		0.489				400
Thalmayer et al. (1964) (Correlation-CP)		0.7295				500
This Work	1.1013	0.4888	-1.579	0.3155	Ag/AgCl	500
Hoover (CV) (2014)	2.52	0.126	-1.502	-0.381	Ag/AgCl	
Hoover et al. (CP) (2014)	1.04	0.672	-1.502	-0.381	Ag/AgCl	

The diffusion coefficient and apparent standard potential of U^{+3}/U and U^{+4}/U^{+3}, based on all references mentioned in Table III.1, are shown in Figs. III.1 and III.2, respectively. To validate Randles-Sevick equation (Eq. (3.1)), cathodic peak current (i_p) versus square root of scan rate is plotted in Fig. III.3 for the 1 wt% of uranium chloride. Fig. III.3 shows that the data for peak A_a, A_c and C_c are linear with R^2 values of 0.9928, 0.9968 and 0.9978, respectively. This implies that Equation (3.1) can be used for calculating the diffusion coefficient at the mentioned peaks.

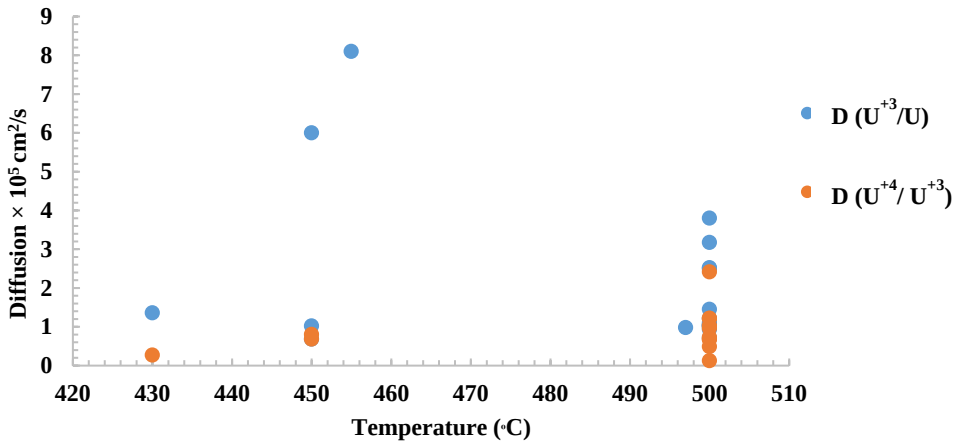

Fig. III.1 Diffusion coefficient values of U^{+3}/U and U^{+4}/U^{+3} reported in Table III.1.

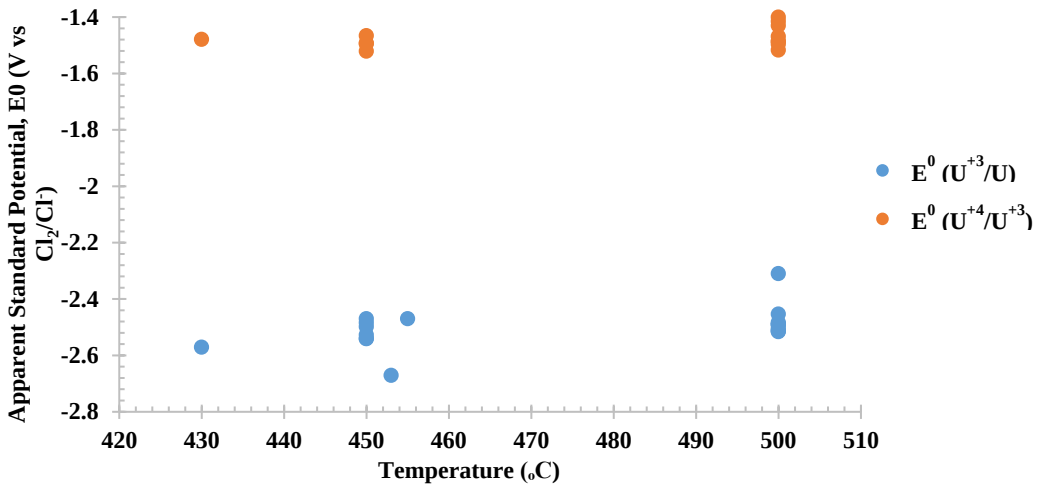

Fig. III.2 Apparent standard potential values of U^{+3}/U and U^{+4}/U^{+3} reported in Table III.1.

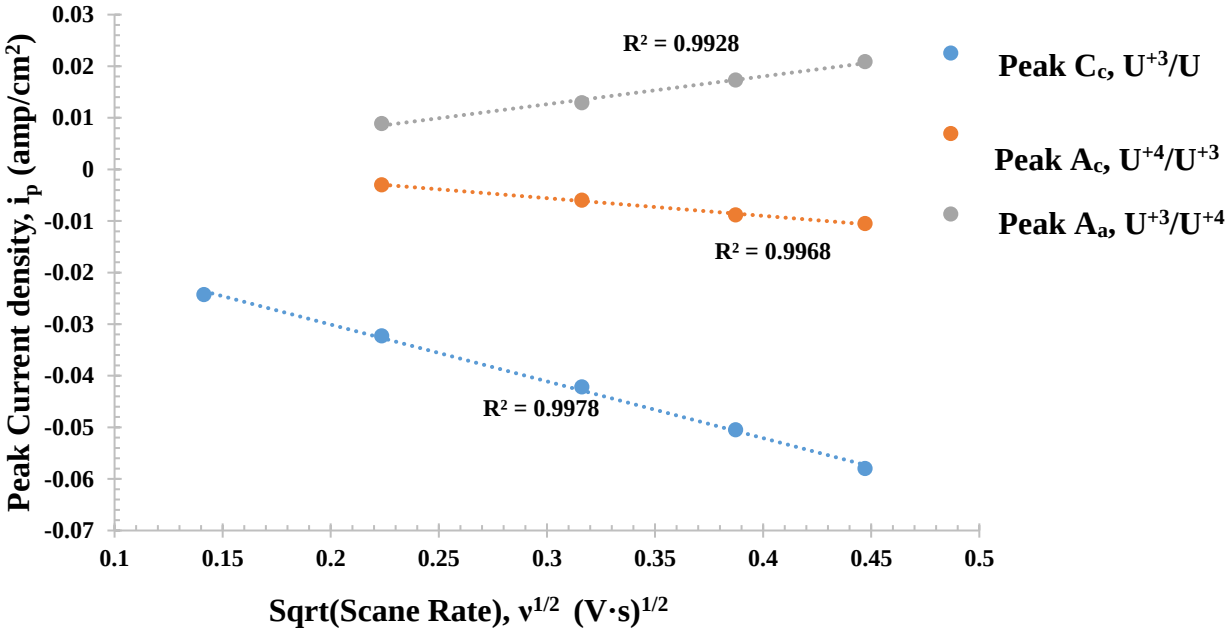

Fig. III. 3 Cathodic peak current vs square root of scan rate for the 1 wt% UCl_3 cyclic voltammogram.

Appendix IV: Diffusion Model for Uranium Chloride

IV.1 *Matlab* Code

Diffusion model *Matlab* code for 1wt% of uranium chloride at 100 mV/s is shown below. To write for the other concentrations and scan rates, values reported in Table 3.2 should be substituted to the **bold** variables.

```
clc
clear

% The Values can be inserted manually at each code or can be asked to put
% by user.

%Do1=input(' Diffusion coefficient of Ox for reversisible part = ');
%DR1=input(' Diffusion coefficient of Red for reversisible part = ');
%Do2=input(' Diffusion coefficient of Ox for irreversisible part = ');
%DR2=input(' Diffusion coefficient of Red for irreversisible part = ');
%n1=input(' number of transfered electron for Reversible part = ');
%n2=input(' number of transfered electron for Irreversible part = ');
%trev1=input(' Irreversible reaction starts at this time = ');
%trev2=input(' Scan is reversed at this time = ');


tinitial=0.0012;
trev1=9.36; % Time (s), Reversible Cathode, reported in Table. 3.2
trev2=23.904; %Irreversible Cathode time+ Reversible Cathode time
landa=trev2;
trev3=trev2+(trev2-trev1);
tfinal=trev3+(trev1-tinitial);

F=96485;
T=773.15; %K
R=8.314;
W=1;
nu=0.1; %Scan rate (V/s)
Ei=-0.003297064;
ks=0.00026;
alpha=0.5;
ii=3; % number of sheet at Excel file which include 1wt% uranium chloride
with 100 mV/s
delta=0.08; %Time interval, Table 3.2

MCl3=344.39; %UCL3
if W==1;
```

```
    A=0.626;
elseif W==2.5;
    A=0.583;
elseif W==5;
    A=0.710;
elseif W==7.5;
    A=0.659;
else W=10;
    A=0.785;
End

RumLiCl=1.502; % g/cm3
RumKCl=1.527;
tmLiCl=610; % C
tmKCl=771; % C
kLiCl=0.000432; %g/cm3C
kKCl=0.000583; %g/cm3C
RuLiCl=RumLiCl-(kLiCl*((T-274.14)-tmLiCl));
RuKCl=RumKCl-(kKCl*((T-274.14)-tmKCl));
Ru=(0.5*RuKCl)+(0.5*RuLiCl);
C=(W/100)*Ru/MCl3; %mol/cm3

%% Reversible Cathodic/Anodic Current and Cathodic Potential:
 % O + ne <---> R
% U(IV) + e <---> U(III)
% Do= D U(IV)/U(III)
% DR= D U(III)/U(IV) %%

% Nicholson & Shain Article:

n1=1; % number of transfer electron at reversible section
Do1=1.05E-05; % Refer to Table .3.2
DR1=3.02E-05; % Refer to Table .3.2
E01=-0.43; %Formal Electrode Potential (Guess) for Cathode side, Table
3.2
E012=-0.25; %Formal Electrode Potential (Guess) for Anodic side, Table
3.2

t=tinitial;
for w=1:round(abs(trev1-t)/delta);
    t=t+delta;
    tetha=exp((n1*F*(Ei-E01)/(R*T)));
    gama=sqrt(Do1/DR1);
    tetgama=tetha*gama;
    M=round((n1*F*nu*t)/(R*T*delta));
    a=(n1*F*nu)/(R*T);
    S1=zeros(M,M);
    for i=1:M;
        j=0;
        B1(i,1)=1/(2*sqrt(delta)*(1+(tetgama*exp(-delta*(i)))));
        while j<M;
            j=j+1;
            N=i;
            if j<N;
                S1(i,j)=sqrt(N-(j-1))-sqrt(N-j);
```

```
                elseif N==j
                    S1(i,j)=1;
                else j>N;
                    S1(i,j)=0;
                end
            end
        end
        x1=S1\B1; % or x=inv(S)*B
        Xat1=x1(i,1);
        I1rev(1,w)=n1*F*A*C*sqrt(Do1*a*pi)*Xat1;
        E1rev(1,w)=E01-
(((R*T)/(n1*F))*log(gama))+(((R*T)/(n1*F))*(log(tetgama)-(a*t)));
    end
    E1=E1rev(1,w);
    t=tinitial;
    for w=1:round(abs(trev1-t)/delta);
        t=t+delta;
        tetha=exp((n1*F*(Ei-E012)/(R*T)));
        gama=sqrt(Do1/DR1);
        tetgama=tetha*gama;
        M=round((n1*F*nu*t)/(R*T*delta));
        a=(n1*F*nu)/(R*T);
        S2=zeros(M,M);
        for i=1:M;
            j=0;
            B2(i,1)=1/(2*sqrt(delta)*(1+(tetgama*exp(-delta*(i)))));
            while j<M;
                j=j+1;
                N=i;
                if j<N;
                    S2(i,j)=sqrt(N-(j-1))-sqrt(N-j);
                elseif N==j
                    S2(i,j)=1;
                else j>N;
                    S2(i,j)=0;
                end
            end
        end
        x2=S2\B2; % or x=inv(S)*B
        Xat2=x2(i,1);
        I12rev(1,w)=n1*F*A*C*sqrt(DR1*a*pi)*Xat2;
        E12rev(1,w)=E012-
(((R*T)/(n1*F))*log(gama))+(((R*T)/(n1*F))*(log(tetgama)-(a*t)));
    end
    E3=E12rev(1,w);

    %% Irreversible Cathodic/Anodic Current:
    % O + ne ----> R      U+3 + 3e -----> U
    % Do=DU(III)/U
    % DR=DU/U(III)

    % Shain & Nicholson for Reversible:

    n2=3; %number of transfer electron at Irreversible section
    DR2=3.8E-04; %Hoover CV 2014, reported in Table. 3.2
    Do2=2E-05; % Refer to Table .3.2
```

```
    E02=-1.61; %Formal Electrode Potential (Guess) at Cathodic side, Table
3.2
    E022=-1.45; %Formal Electrode Potential (Guess) at Anodic side, Table 3.2

    t=trev1;
    for w=1:round(abs(trev2-t)/delta);
        t=t+delta;
        tetha=exp((n2*F*alpha*(Ei-E02)/(R*T)));
        gama=sqrt(Do2/DR2);
        tetgama=tetha*gama;
        M=round((n2*F*nu*alpha*t)/(R*T*delta));
        b=(n2*F*alpha*nu)/(R*T);
        k=ks*exp(((-alpha*n2*F)/(R*T))*(Ei-E02));
        u=log(sqrt(pi*Do2*b)/k);
        S3=zeros(M,M);
        for i=1:M;
            j=0;
            B3(i,1)=1/(2*sqrt(delta)*(1+(tetgama*exp(-delta*(i)))));
            while j<M;
                j=j+1;
                N=i;
                if j<N;
                    S3(i,j)=sqrt(N-(j-1))-sqrt(N-j);
                elseif N==j
                    S3(i,j)=1;
                else j>N;
                    S3(i,j)=0;
                end
            end
        end
        x3=S3\B3; % or x=inv(S)*B
        Xbt3=x3(i,1);
        I1irrev(1,w)=n2*F*A*C*sqrt(Do2*b*pi)*Xbt3;
        E1irrev(1,w)=E02+(((R*T)/(F*n2*alpha))*(u-(b*t)))-
(((R*T)/(F*n2*alpha))*log(sqrt(pi*Do2*b)/ks));
    end
    E2=E1irrev(1,w);

    t=trev1;
    for w=1:round(abs(trev2-t)/(2*delta));
        t=t+delta;
        tetha=exp((n2*F*alpha*(Ei-E022)/(R*T)));
        gama=sqrt(Do2/DR2);
        tetgama=tetha*gama;
        M=round((n2*F*nu*alpha*t)/(R*T*delta));
        b=(n2*F*alpha*nu)/(R*T);
        k=ks*exp(((-alpha*n2*F)/(R*T))*(Ei-E022));
        u=log(sqrt(pi*Do2*b)/k);
        S4=zeros(M,M);
        for i=1:M;
            j=0;
            B4(i,1)=1/(2*sqrt(delta)*(1+(tetgama*exp(-delta*(i)))));
            while j<M;
                j=j+1;
                N=i;
                if j<N;
```

```
                    S4(i,j)=sqrt(N-(j-1))-sqrt(N-j);
                elseif N==j
                    S4(i,j)=1;
                else j>N;
                    S4(i,j)=0;
                end
            end
        end
        x4=S4\B4; % or x=inv(S)*B
        Xbt4=x4(i,1);
        I21irrev(1,w)=n2*F*A*C*sqrt(DR2*b*pi)*Xbt4;
        E21irrev(1,w)=E022+(((R*T)/(F*n2*alpha))*(u-(b*t)))-
(((R*T)/(F*n2*alpha))*log(sqrt(pi*Do2*b)/ks));
    end
    t=trev2;
    for w=1:round(abs(trev3-t)/(2*delta));
        t=t+delta;
        tetha=exp((n2*F*alpha*(Ei-E022)/(R*T)));
        gama=sqrt(Do2/DR2);
        tetgama=tetha*gama;
        M=round((n2*F*nu*alpha*t)/(R*T*delta));
        b=(n2*F*alpha*nu)/(R*T);
        k1=ks*exp(((-alpha*n2*F)/(R*T))*(Ei-E022));
        u1=log(sqrt(pi*Do2*b)/k1);
        S5=zeros(M,M);
        for i=1:M;
            j=0;
            B5(i,1)=1/(2*sqrt(delta)*(1+(tetgama*exp(delta*(i)-
(2*landa*delta)))));
            while j<M;
                j=j+1;
                N=i;
                if j<N;
                    S5(i,j)=sqrt(N-(j-1))-sqrt(N-j);
                elseif N==j
                    S5(i,j)=1;
                else j>N;
                    S5(i,j)=0;
                end
            end
        end
        x5=S5\B5; % or x=inv(S)*B
        Xbt5=x5(i,1);
        I2irrev(1,w)=n2*F*A*C*sqrt(DR2*b*pi)*Xbt5;
        bt=((alpha*n2*F)/(R*T))*((2*nu*landa)-(nu*t));
        E2irrev(1,w)=E022+(((R*T)/(F*n2*alpha))*(u1-(bt)))-
(((R*T)/(F*n2*alpha))*log(sqrt(pi*Do2*b)/ks));
    end

    %% Concentration Plot Cathodic/Anodic side:
    %Part 2: Irreversible Cathodic:
    u=238.0289;
    cl=35.453;
    t=trev1;
    Co=C;
    for w=1:round(abs(trev2-t)/(delta));
```

```
    t=t+delta;
    if w==1;
    t2(1,1)=trev1+delta;
    elseif w>1;
     t2(1,w)=t2(1,w-1)+delta;
    end
    tetha=exp((n2*F*alpha*(Ei-(nu*t)-E02)/(R*T)));
    gama=sqrt(Do2/DR2);
    tetgama=tetha*gama;
    Co1irrev(1,w)=Co*(1-(1/(1+tetgama)));
    CR1irrev(1,w)=Co-Co1irrev(1,w);
end

% %Part 3: Irreversible Anodic:
t=trev2;
CR21=CR1irrev(1,w);
for w=1:round(abs(trev3-t)/(delta));
    t=t+delta;
    if w==1;
    t3(1,w)=trev2+delta;
    elseif w>1;
    t3(1,w)=t3(1,w-1)+delta;
    end
    tetha=exp((n2*F*alpha*(Ei-(2*nu*landa)+(nu*t)-E022)/(R*T)));
    gama=sqrt(Do2/DR2);
    tetgama=tetha*gama;
    CR21irrev(1,w)=CR21*(1-(tetgama/(1+tetgama)));
    Co21irrev(1,w)=CR21-CR21irrev(1,w);
end

%Part 4: Reversible Anodic:
t=trev3;
CR2=Co21irrev(1,w);
for w=1:round(abs(trev3-tfinal)/(delta));
    t=t+delta;
    if w==1;
    t4(1,w)=trev3+delta;
    elseif w>1;
    t4(1,w)=t4(1,w-1)+delta;
    end
    tetha=exp((n1*F*(Ei-(2*nu*landa)+(nu*t)-E012)/(R*T)));
    gama=sqrt(Do1/DR1);
    tetgama=tetha*gama;
    CR2rev(1,w)=CR2*(1-((tetgama/(1+tetgama))));
    Co2rev(1,w)=CR2-CR2rev(1,w);
end

%Part 1: Reversible Cathodic:
t=tinitial;
Co1=Co2rev(1,w);
for w=1:round(abs(trev1-t)/(delta));
    t=t+delta;
    if w==1;
    t1(1,w)=tinitial+delta;
    elseif w>1;
    t1(1,w)=t1(1,w-1)+delta;
```

```
        end
        tetha=exp((n1*F*(Ei-(nu*t)-E01)/(R*T)));
        gama=sqrt(Do1/DR1);
        tetgama=tetha*gama;
        Co1rev(1,w)=Co1*(1-(1/(1+tetgama)));
        CR1rev(1,w)=Co1-Co1rev(1,w);
    end

    %%
    Potential=[E1rev E1irrev E2irrev E21irrev E12rev]';
    Current=[-I1rev -I1irrev I2irrev I21irrev I12rev]';
    Concentration_o_g=[Co1rev*(u+(4*cl)) Co1irrev*(u+(3*cl))
Co21irrev*(u+(3*cl)) Co2rev*(u+(4*cl))]';
    Concentration_R_g=[CR1rev*(u+(4*cl)) CR1irrev*(u+(3*cl))
CR21irrev*(u+(3*cl)) CR2rev*(u+(4*cl))]';
    Concentration_o_mol=[Co1rev Co1irrev Co21irrev Co2rev]';
    Concentration_R_mol=[CR1rev CR1irrev CR21irrev CR2rev]';
    Time=[t1 t2 t3 t4]';
    CV=[Potential Current];
    Concentration=[Time Concentration_o_g Concentration_R_g
Concentration_o_mol Concentration_R_mol];
    file=sprintf('1wt_nu100_CV.csv');
    filename=sprintf('1wt_nu100_Concentration.csv');
    csvwrite(file,CV);
    csvwrite(filename,Concentration);

    %%
    figure % New figure
    plot(t1,Co1rev*(u+(4*cl)),'b*')
    hold on
    plot(t1,CR1rev*(u+(3*cl)),'r--')
    hold on
    plot(t2,Co1irrev*(u+(3*cl)),'r--')
    hold on
    plot(t2,CR1irrev*u,'Ko')
    hold on
    plot(t3,Co21irrev*(u+(3*cl)),'r--')
    hold on
    plot(t3,CR21irrev*u,'Ko')
    hold on
    plot(t4,Co2rev*(u+(4*cl)),'b*')
    hold on
    plot(t4,CR2rev*(u+(3*cl)),'r--')
    hold on
    xlabel('Time (Second)')
    ylabel('Concentration (gr/cm3)')

    %% I vs E Plot and compare with Raw Data
    figure
    plot(E1rev,-I1rev,'b.',E12rev,I12rev,'b.')
    xlabel('Potential Cathodic, E, v')
    ylabel('Current Cathodic, I, amp')
    hold on
    plot(E1irrev,-I1irrev,'r*',E21irrev,I21irrev,'r*',E2irrev,I2irrev,'r*')
    hold on
    sheetii=xlsread('rawdata-U-1.xlsx',ii);
```

```
    plot(sheetii(:,1),sheetii(:,2),'K-');
    hold on

    %% Calculating Concentration:
    m=input(' Input the number of concentration points =');
    [E,I]=ginput(m);

tc1(m,1)=0;CO1Re(m,1)=0;CR1Re(m,1)=0;tc2(m,1)=0;CO1Ire(m,1)=0;CR1Ire(m,1)=0;

tc3(m,1)=0;CO21Ire(m,1)=0;CR21Ire(m,1)=0;tc4(m,1)=0;CO2Re(m,1)=0;CR2Re(m,1)=0
;
    for i=1:m
        if  abs(E(i,1))<abs(E1) & I(i,1)<0 %Part1, Reversible Cathodic
            X=E(i,1);
            tetha1=exp((n1*F*(Ei-E01)/(R*T)));
            tetgama1=tetha1*gama;
            gama=sqrt(Do1/DR1);
            a=(n1*F*nu)/(R*T);
            tc1(i,1)=(log(tetgama1)+(((n1*F)/(R*T))*(E01-X))-log(gama))/a;
            tetha=exp((n1*F*(Ei-(nu*tc1(i,1))-E01)/(R*T)));
            tetgama=tetha*gama;
            CO1Re(i,1)=Co1*(1-(1/(1+tetgama)));
            CR1Re(i,1)=Co1-CO1Re(i,1);
            elseif  abs(E1)<abs(E(i,1)) & abs(E(i,1))<abs(E2) &
I(i,1)<0 %Part2, Irreversible Cathodic
            X=E(i,1);
            tetha1=exp((n2*F*alpha*(Ei-E02)/(R*T)));
            gama=sqrt(Do2/DR2);
            tetgama1=tetha1*gama;
            b=(n2*F*alpha*nu)/(R*T);
            k=ks*exp(((-alpha*n2*F)/(R*T))*(Ei-E02));
            u=log(sqrt(pi*Do2*b)/k);
            tc2(i,1)=(u-(((alpha*n2*F)/(R*T))*(X-E02))-
log(sqrt(pi*Do2*b)/ks))/b;
            tetha=exp((n2*F*alpha*(Ei-(nu*tc2(i,1))-E02)/(R*T)));
            tetgama=tetha*gama;
            CO1Ire(i,1)=Co*(1-(1/(1+tetgama)));
            CR1Ire(i,1)=Co-CO1Ire(i,1);
            elseif abs(E(i,1))<abs(E3) & I(i,1)>0  %Part4, Reversible Anodic
            X=E(i,1);
            tetha1=exp((n1*F*(Ei-E012)/(R*T)));
            tetgama1=tetha1*gama;
            gama=sqrt(Do1/DR1);
            a=(n1*F*nu)/(R*T);
            tc4(i,1)=trev2+(trev2-(log(tetgama1)+(((n1*F)/(R*T))*(E012-X))-
log(gama)/a));
            tetha=exp((n1*F*(Ei-(2*nu*landa)+(nu*tc4(i,1))-E012)/(R*T)));
            tetgama=tetha*gama;
            CR2Re(i,1)=CR2*(1-((tetgama/(1+tetgama))));
            CO2Re(i,1)=CR2-CR2Re(i,1);
            else %Part3, Irreversible Anodic
            X=E(i,1);
            tetha1=exp((n2*F*alpha*(Ei-E022)/(R*T)));
            gama=sqrt(Do2/DR2);
            tetgama1=tetha1*gama;
            b=(n2*F*alpha*nu)/(R*T);
```

```
            k=ks*exp(((-alpha*n2*F)/(R*T))*(Ei-E022));
            u=log(sqrt(pi*Do2*b)/k);
            tc3(i,1)=trev2+(trev2-(u-(((alpha*n2*F)/(R*T))*(X-E022))-
log(sqrt(pi*Do2*b)/ks))/b);
            tetha=exp((n2*F*alpha*(Ei-(2*nu*landa)+(nu*tc3(i,1))-
E022)/(R*T)));
            tetgama=tetha*gama;
            CR21Ire(i,1)=CR21*(1-(tetgama/(1+tetgama)));
            CO21Ire(i,1)=CR21-CR21Ire(i,1);
        end

    end
    Time_Second=tc1+tc2+tc3+tc4;
    Concentration_O=CO1Re+CO1Ire+CO21Ire+CO2Re;
    Concentration_R=CR1Re+CR1Ire+CR21Ire+CR2Re;
    R=strsplit(num2str(1:m),' ');
    Table=table(Time_Second,Concentration_R,Concentration_O,...
        'RowNames',R)
```

IV.2 GUI Code

```
    function varargout = GUI5final(varargin)
    % GUI5final MATLAB code for GUI5final.fig
    %      GUI5final, by itself, creates a new GUI5final or raises the
existing
    %      singleton*.
    %
    %      H = GUI5final returns the handle to a new GUI5final or the handle
to
    %      the existing singleton*.
    %
    %      GUI5final('CALLBACK',hObject,eventData,handles,...) calls the
local
    %      function named CALLBACK in GUI5final.M with the given input
arguments.
    %
    %      GUI5final('Property','Value',...) creates a new GUI5final or
raises the
    %      existing singleton*.  Starting from the left, property value pairs
are
    %      applied to the GUI before GUI5final_OpeningFcn gets called.  An
    %      unrecognized property name or invalid value makes property
application
    %      stop.  All inputs are passed to GUI5final_OpeningFcn via varargin.
    %
    %      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only
one
    %      instance to run (singleton)".
    %
    % See also: GUIDE, GUIDATA, GUIHANDLES

    % Edit the above text to modify the response to help GUI5final

    % Last Modified by GUIDE v2.5 26-Dec-2015 13:46:51
```

```
% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @GUI5final_OpeningFcn, ...
                   'gui_OutputFcn',  @GUI5final_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before GUI5final is made visible.
function GUI5final_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to GUI5final (see VARARGIN)

% Choose default command line output for GUI5final
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes GUI5final wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = GUI5final_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on selection change in popupconcentration.
function popupconcentration_Callback(hObject, eventdata, handles)
% hObject    handle to popupconcentration (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
    % handles    structure with handles and user data (see GUIDATA)

    % Hints: contents = cellstr(get(hObject,'String')) returns
popupconcentration contents as cell array
    %        contents{get(hObject,'Value')} returns selected item from
popupconcentration
    % Determine the selected data set.
    str1 = get(hObject, 'String');
    val1 = get(hObject,'Value');
    % Set current data to the selected data set.
    switch str1{val1};
    case '1 wt%'
        handles.W = 1;
            handles.A = 0.626;
        case '2.5 wt%'
             handles.W = 2.5;
                handles.A = 0.583;
            case '5 wt%'
                 handles.W = 5;
                     handles.A = 0.710;
                case '7.5 wt%'
                     handles.W = 7.5;
                         handles.A = 0.659;
                    case '10 wt%'
                         handles.W = 10;
                             handles.A = 0.785;
    end
    % Save the handles structure.
    guidata(hObject,handles);


    % --- Executes during object creation, after setting all properties.
    function popupconcentration_CreateFcn(hObject, eventdata, handles)
    % hObject    handle to popupconcentration (see GCBO)
    % eventdata  reserved - to be defined in a future version of MATLAB
    % handles    empty - handles not created until after all CreateFcns
called

    % Hint: popupmenu controls usually have a white background on Windows.
    %       See ISPC and COMPUTER.
    if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
        set(hObject,'BackgroundColor','white');
    end


     function edit1_Callback(hObject, eventdata, handles)
    % hObject    handle to edit1 (see GCBO)
    % eventdata  reserved - to be defined in a future version of MATLAB
    % handles    structure with handles and user data (see GUIDATA)

    % Hints: get(hObject,'String') returns contents of edit1 as text
    %        str2double(get(hObject,'String')) returns contents of edit1 as a
double
    str2 = get(hObject, 'String');
```

```
    val2 = str2num(str2);
    handles.nu= val2;
    guidata(hObject,handles);

    % --- Executes during object creation, after setting all properties.
    function edit1_CreateFcn(hObject, eventdata, handles)
    % hObject    handle to edit1 (see GCBO)
    % eventdata  reserved - to be defined in a future version of MATLAB
    % handles    empty - handles not created until after all CreateFcns
called

    % Hint: edit controls usually have a white background on Windows.
    %       See ISPC and COMPUTER.
    if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
        set(hObject,'BackgroundColor','white');
    end


    % --- Executes during object creation, after setting all properties.
    function axes2_CreateFcn(hObject, eventdata, handles)
    % hObject    handle to axes2 (see GCBO)
    % eventdata  reserved - to be defined in a future version of MATLAB
    % handles    empty - handles not created until after all CreateFcns
called

    % Hint: place code in OpeningFcn to populate axes2


    % --- Executes on button press in pushbutton1.
    function pushbutton1_Callback(hObject, eventdata, handles)
    % hObject    handle to pushbutton1 (see GCBO)
    % eventdata  reserved - to be defined in a future version of MATLAB
    % handles    structure with handles and user data (see GUIDATA)
    if handles.W == 1 && handles.nu == 0.1
    handles.Do1 = 1.05E-05;
    handles.DR1 = 3.02E-05;
    handles.E01 = -0.43;
    handles.E012 = -0.25;
    handles.Do2 = 2E-05;
    handles.DR2 = 3.8E-04;
    handles.E02 = -1.61;
    handles.E022 = -1.45;
    handles.tinitial = 0.0012;
    handles.trev1 = 9.36;
    handles.trev2 = 23.904;
    handles.delta = 0.08;
    elseif handles.W==1 && 0.1<handles.nu &&handles.nu<0.15
    handles.nu1=0.1;
    handles.Do1_1 = 1.05E-05;
    handles.DR1_1 = 3.02E-05;
    handles.E01_1 = -0.43;
    handles.E012_1 = -0.25;
    handles.Do2_1 = 2E-05;
    handles.DR2_1 = 3.8E-04;
```

```
    handles.E02_1 = -1.61;
    handles.E022_1 = -1.45;
    handles.tinitial_1 = 0.0012;
    handles.trev1_1 = 9.36;
    handles.trev2_1 = 23.904;
    handles.delta_1 = 0.08;

    handles.nu2=0.15;
    handles.Do1_2 = 1.05E-05;
    handles.DR1_2 = 3.02E-05;
    handles.E01_2 = -0.43;
    handles.E012_2 = -0.25;
    handles.Do2_2 = 2E-05;
    handles.DR2_2 = 2.9E-04;
    handles.E02_2 = -1.61;
    handles.E022_2 = -1.42;
    handles.tinitial_2 = 0.0012;
    handles.trev1_2 = 6.624;
    handles.trev2_2 = 15.829;
    handles.delta_2 = 0.08;

    handles.Do1=handles.Do1_1+(((handles.nu-handles.nu1)*(handles.Do1_2-
handles.Do1_1))/(handles.nu2-handles.nu1));
    handles.DR1=handles.DR1_1+(((handles.nu-handles.nu1)*(handles.DR1_2-
handles.DR1_1))/(handles.nu2-handles.nu1));
    handles.E01=handles.E01_1+(((handles.nu-handles.nu1)*(handles.E01_2-
handles.E01_1))/(handles.nu2-handles.nu1));
    handles.E012=handles.E012_1+(((handles.nu-handles.nu1)*(handles.E012_2-
handles.E012_1))/(handles.nu2-handles.nu1));

    handles.Do2=handles.Do2_1+(((handles.nu-handles.nu1)*(handles.Do2_2-
handles.Do2_1))/(handles.nu2-handles.nu1));
    handles.DR2=handles.DR2_1+(((handles.nu-handles.nu1)*(handles.DR2_2-
handles.DR2_1))/(handles.nu2-handles.nu1));
    handles.E02=handles.E02_1+(((handles.nu-handles.nu1)*(handles.E02_2-
handles.E02_1))/(handles.nu2-handles.nu1));
    handles.E022=handles.E022_1+(((handles.nu-handles.nu1)*(handles.E022_2-
handles.E022_1))/(handles.nu2-handles.nu1));

    handles.trev1=handles.trev1_1+(((handles.nu-
handles.nu1)*(handles.trev1_2-handles.trev1_1))/(handles.nu2-handles.nu1));
    handles.trev2=handles.trev2_1+(((handles.nu-
handles.nu1)*(handles.trev2_2-handles.trev2_1))/(handles.nu2-handles.nu1));
    handles.tinitial=handles.tinitial_1+(((handles.nu-
handles.nu1)*(handles.tinitial_2-handles.tinitial_1))/(handles.nu2-
handles.nu1));
    handles.delta=handles.delta_1+(((handles.nu-
handles.nu1)*(handles.delta_2-handles.delta_1))/(handles.nu2-handles.nu1));
    elseif handles.W == 1 && handles.nu == 0.15
    handles.Do1 = 1.05E-05;
    handles.DR1 = 3.02E-05;
    handles.E01 = -0.43;
    handles.E012 = -0.25;
    handles.Do2 = 2E-05;
    handles.handles.DR2 = 2.9E-04;
    handles.E02 = -1.61;
```

```
    handles.E022 = -1.42;
    handles.tinitial = 0.012;
    handles.trev1 = 6.624;
    handles.trev2 = 15.829;
    handles.delta = 0.08;
    elseif handles.W== 1 && 0.15<handles.nu && handles.nu<0.2
    handles.nu1=0.15;
    handles.Do1_1 = 1.05E-05;
    handles.DR1_1 = 3.02E-05;
    handles.E01_1 = -0.43;
    handles.E012_1 = -0.25;
    handles.Do2_1 = 2E-05;
    handles.DR2_1 = 2.9E-04;
    handles.E02_1 = -1.61;
    handles.E022_1 = -1.42;
    handles.tinitial_1 = 0.0012;
    handles.trev1_1 = 6.624;
    handles.trev2_1 = 15.829;
    handles.delta_1 = 0.08;

     handles.nu2=0.2;
     handles.Do1_2 = 1.05E-05;
     handles.DR1_2 = 3.02E-05;
     handles.E01_2 = -0.43;
     handles.E012_2 = -0.25;
     handles.Do2_2 = 2E-05;
     handles.DR2_2 = 2.5E-04;
     handles.E02_2 = -1.61;
     handles.E022_2 = -1.41;
     handles.tinitial_2 = 0.012;
     handles.trev1_2 = 4.764;
     handles.trev2_2 = 12.08;
     handles.delta_2 = 0.08;

    handles.Do1=handles.Do1_1+(((handles.nu-handles.nu1)*(handles.Do1_2-
handles.Do1_1))/(handles.nu2-handles.nu1));
    handles.DR1=handles.DR1_1+(((handles.nu-handles.nu1)*(handles.DR1_2-
handles.DR1_1))/(handles.nu2-handles.nu1));
    handles.E01=handles.E01_1+(((handles.nu-handles.nu1)*(handles.E01_2-
handles.E01_1))/(handles.nu2-handles.nu1));
    handles.E012=handles.E012_1+(((handles.nu-handles.nu1)*(handles.E012_2-
handles.E012_1))/(handles.nu2-handles.nu1));

    handles.Do2=handles.Do2_1+(((handles.nu-handles.nu1)*(handles.Do2_2-
handles.Do2_1))/(handles.nu2-handles.nu1));
    handles.DR2=handles.DR2_1+(((handles.nu-handles.nu1)*(handles.DR2_2-
handles.DR2_1))/(handles.nu2-handles.nu1));
    handles.E02=handles.E02_1+(((handles.nu-handles.nu1)*(handles.E02_2-
handles.E02_1))/(handles.nu2-handles.nu1));
    handles.E022=handles.E022_1+(((handles.nu-handles.nu1)*(handles.E022_2-
handles.E022_1))/(handles.nu2-handles.nu1));

    handles.trev1=handles.trev1_1+(((handles.nu-
handles.nu1)*(handles.trev1_2-handles.trev1_1))/(handles.nu2-handles.nu1));
    handles.trev2=handles.trev2_1+(((handles.nu-
handles.nu1)*(handles.trev2_2-handles.trev2_1))/(handles.nu2-handles.nu1));
```

```
    handles.tinitial=handles.tinitial_1+(((handles.nu-
handles.nu1)*(handles.tinitial_2-handles.tinitial_1))/(handles.nu2-
handles.nu1));
    handles.delta=handles.delta_1+(((handles.nu-
handles.nu1)*(handles.delta_2-handles.delta_1))/(handles.nu2-handles.nu1));
    elseif handles.W == 1 && handles.nu == 0.2
     handles.Do1 = 1.05E-05;
     handles.DR1 = 3.02E-05;
     handles.E01 = -0.43;
     handles.E012 = -0.25;
     handles.Do2 = 2E-05;
     handles.DR2 = 2.5E-04;
     handles.E02 = -1.61;
     handles.E022 = -1.41;
     handles.tinitial = 0.012;
     handles.trev1 = 4.764;
     handles.trev2 = 12.08;
    handles. delta = 0.08;
    elseif handles.W== 1 && handles.nu>0.2
    handles.nu1=0.15;
    handles.Do1_1 = 1.05E-05;
    handles.DR1_1 = 3.02E-05;
    handles.E01_1 = -0.43;
    handles.E012_1 = -0.25;
    handles.Do2_1 = 2E-05;
    handles.DR2_1 = 2.9E-04;
    handles.E02_1 = -1.61;
    handles.E022_1 = -1.42;
    handles.tinitial_1 = 0.0012;
    handles.trev1_1 = 6.624;
    handles.trev2_1 = 15.829;
    handles.delta_1 = 0.08;

    handles.nu2=0.2;
     handles.Do1_2 = 1.05E-05;
     handles.DR1_2 = 3.02E-05;
     handles.E01_2 = -0.43;
     handles.E012_2 = -0.25;
     handles.Do2_2 = 2E-05;
     handles.DR2_2 = 2.5E-04;
     handles.E02_2 = -1.61;
     handles.E022_2 = -1.41;
     handles.tinitial_2 = 0.012;
     handles.trev1_2 = 4.764;
     handles.trev2_2 = 12.08;
     handles.delta_2 = 0.08;

    handles.Do1=handles.Do1_1+(((handles.nu-handles.nu1)*(handles.Do1_2-
handles.Do1_1))/(handles.nu2-handles.nu1));
    handles.DR1=handles.DR1_1+(((handles.nu-handles.nu1)*(handles.DR1_2-
handles.DR1_1))/(handles.nu2-handles.nu1));
    handles.E01=handles.E01_1+(((handles.nu-handles.nu1)*(handles.E01_2-
handles.E01_1))/(handles.nu2-handles.nu1));
    handles.E012=handles.E012_1+(((handles.nu-handles.nu1)*(handles.E012_2-
handles.E012_1))/(handles.nu2-handles.nu1));
```

```
    handles.Do2=handles.Do2_1+(((handles.nu-handles.nu1)*(handles.Do2_2-
handles.Do2_1))/(handles.nu2-handles.nu1));
    handles.DR2=handles.DR2_1+(((handles.nu-handles.nu1)*(handles.DR2_2-
handles.DR2_1))/(handles.nu2-handles.nu1));
    handles.E02=handles.E02_1+(((handles.nu-handles.nu1)*(handles.E02_2-
handles.E02_1))/(handles.nu2-handles.nu1));
    handles.E022=handles.E022_1+(((handles.nu-handles.nu1)*(handles.E022_2-
handles.E022_1))/(handles.nu2-handles.nu1));

    handles.trev1=handles.trev1_1+(((handles.nu-
handles.nu1)*(handles.trev1_2-handles.trev1_1))/(handles.nu2-handles.nu1));
    handles.trev2=handles.trev2_1+(((handles.nu-
handles.nu1)*(handles.trev2_2-handles.trev2_1))/(handles.nu2-handles.nu1));
    handles.tinitial=handles.tinitial_1+(((handles.nu-
handles.nu1)*(handles.tinitial_2-handles.tinitial_1))/(handles.nu2-
handles.nu1));
    handles.delta=handles.delta_1+(((handles.nu-
handles.nu1)*(handles.delta_2-handles.delta_1))/(handles.nu2-handles.nu1));
    elseif  handles.W == 2.5 &&  handles.nu == 0.1
         handles.Do1 = 0.7E-05;
         handles.DR1 = 3.02E-05;
         handles.E01 = -0.43;
         handles.E012 = -0.25;
         handles.Do2 = 2E-05;
         handles.DR2 = 3E-04;
         handles.E02 = -1.6;
         handles.E022 = -1.34;
         handles.tinitial = 0.0012;
         handles.trev1 = 9.744;
         handles.trev2 = 24.045;
         handles.delta = 0.08;
    elseif handles.W==2.5 && 0.1<handles.nu && handles.nu<0.15
         handles.nu1=0.1;
         handles.Do1_1 = 0.7E-05;
         handles.DR1_1 = 3.02E-05;
         handles.E01_1 = -0.43;
         handles.E012_1 = -0.25;
         handles.Do2_1 = 2E-05;
         handles.DR2_1 = 3E-04;
         handles.E02_1 = -1.6;
         handles.E022_1 = -1.34;
         handles.tinitial_1 = 0.0012;
         handles.trev1_1 = 9.744;
         handles.trev2_1= 24.045;
         handles.delta_1= 0.08;

         handles.nu2=0.15;
         handles.Do1_2 = 0.7E-05;
         handles.DR1_2 = 2.7E-05;
         handles.E01_2 = -0.43;
         handles.E012_2 = -0.25;
         handles.Do2_2 = 2E-05;
         handles.DR2_2 = 2.15E-04;
         handles.E02_2 = -1.6;
         handles.E022_2 = -1.34;
         handles.tinitial_2 = 0.0012;
```

```
        handles.trev1_2 = 6.672;
        handles.trev2_2 = 16;
        handles.delta_2 = 0.08;

    handles.Do1=handles.Do1_1+(((handles.nu-handles.nu1)*(handles.Do1_2-
handles.Do1_1))/(handles.nu2-handles.nu1));
    handles.DR1=handles.DR1_1+(((handles.nu-handles.nu1)*(handles.DR1_2-
handles.DR1_1))/(handles.nu2-handles.nu1));
    handles.E01=handles.E01_1+(((handles.nu-handles.nu1)*(handles.E01_2-
handles.E01_1))/(handles.nu2-handles.nu1));
    handles.E012=handles.E012_1+(((handles.nu-handles.nu1)*(handles.E012_2-
handles.E012_1))/(handles.nu2-handles.nu1));

    handles.Do2=handles.Do2_1+(((handles.nu-handles.nu1)*(handles.Do2_2-
handles.Do2_1))/(handles.nu2-handles.nu1));
    handles.DR2=handles.DR2_1+(((handles.nu-handles.nu1)*(handles.DR2_2-
handles.DR2_1))/(handles.nu2-handles.nu1));
    handles.E02=handles.E02_1+(((handles.nu-handles.nu1)*(handles.E02_2-
handles.E02_1))/(handles.nu2-handles.nu1));
    handles.E022=handles.E022_1+(((handles.nu-handles.nu1)*(handles.E022_2-
handles.E022_1))/(handles.nu2-handles.nu1));

    handles.trev1=handles.trev1_1+(((handles.nu-
handles.nu1)*(handles.trev1_2-handles.trev1_1))/(handles.nu2-handles.nu1));
    handles.trev2=handles.trev2_1+(((handles.nu-
handles.nu1)*(handles.trev2_2-handles.trev2_1))/(handles.nu2-handles.nu1));
    handles.tinitial=handles.tinitial_1+(((handles.nu-
handles.nu1)*(handles.tinitial_2-handles.tinitial_1))/(handles.nu2-
handles.nu1));
    handles.delta=handles.delta_1+(((handles.nu-
handles.nu1)*(handles.delta_2-handles.delta_1))/(handles.nu2-handles.nu1));
    elseif  handles.W == 2.5 &&  handles.nu == 0.15
        handles.Do1 = 0.7E-05;
        handles.DR1 = 2.7E-05;
        handles.E01 = -0.43;
        handles.E012 = -0.25;
        handles.Do2 = 2E-05;
        handles.DR2 = 2.15E-04;
        handles. E02 = -1.6;
        handles.E022 = -1.34;
        handles.tinitial = 0.0012;
        handles.trev1 = 6.672;
        handles.trev2 = 16;
        handles.delta = 0.08;
    elseif handles.W==2.5 && 0.15<handles.nu && handles.nu<0.2
        handles.nu1=0.15;
        handles.Do1_1 = 0.7E-05;
        handles.DR1_1 = 2.7E-05;
        handles.E01_1 = -0.43;
        handles.E012_1 = -0.25;
        handles.Do2_1 = 2E-05;
        handles.DR2_1 = 2.15E-04;
        handles.E02_1 = -1.6;
        handles.E022_1 = -1.34;
        handles.tinitial_1 = 0.0012;
        handles.trev1_1 = 6.672;
```

```
        handles.trev2_1 = 16;
        handles.delta_1 = 0.08;

        handles.nu2=0.2;
        handles.Do1_2 = 0.7E-05;
        handles.DR1_2 = 3.02E-05;
        handles.E01_2 = -0.43;
        handles.E012_2 = -0.25;
        handles.Do2_2 = 1.82E-05;
        handles.DR2_2 = 1.65E-04;
        handles.E02_2 = -1.61;
        handles.E022_2 = -1.34;
        handles.tinitial_2 = 0.0012;
        handles.trev1_2 = 4.908;
        handles.trev2_2 = 11.976;
        handles.delta_2 = 0.08;

    handles.Do1=handles.Do1_1+(((handles.nu-handles.nu1)*(handles.Do1_2-
handles.Do1_1))/(handles.nu2-handles.nu1));
    handles.DR1=handles.DR1_1+(((handles.nu-handles.nu1)*(handles.DR1_2-
handles.DR1_1))/(handles.nu2-handles.nu1));
    handles.E01=handles.E01_1+(((handles.nu-handles.nu1)*(handles.E01_2-
handles.E01_1))/(handles.nu2-handles.nu1));
    handles.E012=handles.E012_1+(((handles.nu-handles.nu1)*(handles.E012_2-
handles.E012_1))/(handles.nu2-handles.nu1));

    handles.Do2=handles.Do2_1+(((handles.nu-handles.nu1)*(handles.Do2_2-
handles.Do2_1))/(handles.nu2-handles.nu1));
    handles.DR2=handles.DR2_1+(((handles.nu-handles.nu1)*(handles.DR2_2-
handles.DR2_1))/(handles.nu2-handles.nu1));
    handles.E02=handles.E02_1+(((handles.nu-handles.nu1)*(handles.E02_2-
handles.E02_1))/(handles.nu2-handles.nu1));
    handles.E022=handles.E022_1+(((handles.nu-handles.nu1)*(handles.E022_2-
handles.E022_1))/(handles.nu2-handles.nu1));

    handles.trev1=handles.trev1_1+(((handles.nu-
handles.nu1)*(handles.trev1_2-handles.trev1_1))/(handles.nu2-handles.nu1));
    handles.trev2=handles.trev2_1+(((handles.nu-
handles.nu1)*(handles.trev2_2-handles.trev2_1))/(handles.nu2-handles.nu1));
    handles.tinitial=handles.tinitial_1+(((handles.nu-
handles.nu1)*(handles.tinitial_2-handles.tinitial_1))/(handles.nu2-
handles.nu1));
    handles.delta=handles.delta_1+(((handles.nu-
handles.nu1)*(handles.delta_2-handles.delta_1))/(handles.nu2-handles.nu1));
    elseif  handles.W == 2.5 &&  handles.nu == 0.2
        handles.Do1 = 0.7E-05;
        handles.DR1 = 3.02E-05;
        handles.E01 = -0.43;
        handles.E012 = -0.25;
        handles.Do2 = 1.82E-05;
        handles.DR2 = 1.65E-04;
        handles.E02 = -1.61;
        handles.E022 = -1.34;
        handles.tinitial = 0.0012;
        handles.trev1 = 3.996;
        handles.trev2 = 12.108;
```

```
        handles.delta = 0.08;
    elseif handles.W==2.5 && 0.2<handles.nu && handles.nu<0.3
        handles.nu1=0.2;
        handles.Do1_1 = 0.7E-05;
        handles.DR1_1 = 3.02E-05;
        handles.E01_1 = -0.43;
        handles.E012_1 = -0.25;
        handles.Do2_1 = 1.82E-05;
        handles.DR2_1 = 1.65E-04;
        handles.E02_1 = -1.61;
        handles.E022_1 = -1.34;
        handles.tinitial_1 = 0.0012;
        handles.trev1_1 = 3.996;
        handles.trev2_1 = 12.108;
        handles.delta_1 = 0.08;

        handles.nu2=0.3;
        handles.Do1_2 = 1.05E-05;
        handles.DR1_2 = 3.02E-05;
        handles.E01_2 = -0.43;
        handles.E012_2 = -0.25;
        handles.Do2_2 = 1.75E-05;
        handles.DR2_2 = 1.3E-04;
        handles.E02_2 = -1.62;
        handles.E022_2 = -1.32;
        handles.tinitial_2 = 0.0012;
        handles.trev1_2 = 3.16;
        handles.trev2_2 = 8;
        handles.delta_2 = 0.08;

    handles.Do1=handles.Do1_1+(((handles.nu-handles.nu1)*(handles.Do1_2-
handles.Do1_1))/(handles.nu2-handles.nu1));
    handles.DR1=handles.DR1_1+(((handles.nu-handles.nu1)*(handles.DR1_2-
handles.DR1_1))/(handles.nu2-handles.nu1));
    handles.E01=handles.E01_1+(((handles.nu-handles.nu1)*(handles.E01_2-
handles.E01_1))/(handles.nu2-handles.nu1));
    handles.E012=handles.E012_1+(((handles.nu-handles.nu1)*(handles.E012_2-
handles.E012_1))/(handles.nu2-handles.nu1));

    handles.Do2=handles.Do2_1+(((handles.nu-handles.nu1)*(handles.Do2_2-
handles.Do2_1))/(handles.nu2-handles.nu1));
    handles.DR2=handles.DR2_1+(((handles.nu-handles.nu1)*(handles.DR2_2-
handles.DR2_1))/(handles.nu2-handles.nu1));
    handles.E02=handles.E02_1+(((handles.nu-handles.nu1)*(handles.E02_2-
handles.E02_1))/(handles.nu2-handles.nu1));
    handles.E022=handles.E022_1+(((handles.nu-handles.nu1)*(handles.E022_2-
handles.E022_1))/(handles.nu2-handles.nu1));

    handles.trev1=handles.trev1_1+(((handles.nu-
handles.nu1)*(handles.trev1_2-handles.trev1_1))/(handles.nu2-handles.nu1));
    handles.trev2=handles.trev2_1+(((handles.nu-
handles.nu1)*(handles.trev2_2-handles.trev2_1))/(handles.nu2-handles.nu1));
    handles.tinitial=handles.tinitial_1+(((handles.nu-
handles.nu1)*(handles.tinitial_2-handles.tinitial_1))/(handles.nu2-
handles.nu1));
```

```
    handles.delta=handles.delta_1+(((handles.nu-
handles.nu1)*(handles.delta_2-handles.delta_1))/(handles.nu2-handles.nu1));
    elseif  handles.W == 2.5 &&  handles.nu == 0.3
         handles.Do1 = 1.05E-05;
         handles.DR1 = 3.02E-05;
         handles.E01 = -0.43;
         handles.E012 = -0.25;
         handles.Do2 = 1.75E-05;
         handles.DR2 = 1.3E-04;
         handles.E02 = -1.62;
         handles.handles.E022 = -1.32;
         handles.tinitial = 0.0012;
         handles.trev1 = 3.16;
         handles.trev2 = 8;
         handles.delta = 0.08;
    elseif handles.W==2.5 &&  handles.nu>0.3
         handles.nu1=0.2;
         handles.Do1_1 = 0.7E-05;
         handles.DR1_1 = 3.02E-05;
         handles.E01_1 = -0.43;
         handles.E012_1 = -0.25;
         handles.Do2_1 = 1.82E-05;
         handles.DR2_1 = 1.65E-04;
         handles.E02_1 = -1.61;
         handles.E022_1 = -1.34;
         handles.tinitial_1 = 0.0012;
         handles.trev1_1 = 4.908;
         handles.trev2_1 = 11.976;
         handles.delta_1 = 0.08;

         handles.nu2=0.3;
         handles.Do1_2 = 1.05E-05;
         handles.DR1_2 = 3.02E-05;
         handles.E01_2 = -0.43;
         handles.E012_2 = -0.25;
         handles.Do2_2 = 1.75E-05;
         handles.DR2_2 = 1.3E-04;
         handles.E02_2 = -1.62;
         handles.E022_2 = -1.32;
         handles.tinitial_2 = 0.0012;
         handles.trev1_2 = 3.16;
         handles.trev2_2 = 8;
         handles.delta_2 = 0.08;

    handles.Do1=handles.Do1_1+(((handles.nu-handles.nu1)*(handles.Do1_2-
handles.Do1_1))/(handles.nu2-handles.nu1));
    handles.DR1=handles.DR1_1+(((handles.nu-handles.nu1)*(handles.DR1_2-
handles.DR1_1))/(handles.nu2-handles.nu1));
    handles.E01=handles.E01_1+(((handles.nu-handles.nu1)*(handles.E01_2-
handles.E01_1))/(handles.nu2-handles.nu1));
    handles.E012=handles.E012_1+(((handles.nu-handles.nu1)*(handles.E012_2-
handles.E012_1))/(handles.nu2-handles.nu1));

    handles.Do2=handles.Do2_1+(((handles.nu-handles.nu1)*(handles.Do2_2-
handles.Do2_1))/(handles.nu2-handles.nu1));
```

```
    handles.DR2=handles.DR2_1+(((handles.nu-handles.nu1)*(handles.DR2_2-
handles.DR2_1))/(handles.nu2-handles.nu1));
    handles.E02=handles.E02_1+(((handles.nu-handles.nu1)*(handles.E02_2-
handles.E02_1))/(handles.nu2-handles.nu1));
    handles.E022=handles.E022_1+(((handles.nu-handles.nu1)*(handles.E022_2-
handles.E022_1))/(handles.nu2-handles.nu1));

    handles.trev1=handles.trev1_1+(((handles.nu-
handles.nu1)*(handles.trev1_2-handles.trev1_1))/(handles.nu2-handles.nu1));
    handles.trev2=handles.trev2_1+(((handles.nu-
handles.nu1)*(handles.trev2_2-handles.trev2_1))/(handles.nu2-handles.nu1));
    handles.tinitial=handles.tinitial_1+(((handles.nu-
handles.nu1)*(handles.tinitial_2-handles.tinitial_1))/(handles.nu2-
handles.nu1));
    handles.delta=handles.delta_1+(((handles.nu-
handles.nu1)*(handles.delta_2-handles.delta_1))/(handles.nu2-handles.nu1));
    elseif handles.W == 5 && handles.nu == 0.25
        handles.Do1 = 0.4E-05;
        handles.DR1 = 2.2E-05;
        handles.E01 = -0.430;
        handles. E012 = -0.26;
        handles.Do2 = 1.50E-05;
        handles.DR2 = 1.35E-04;
        handles.E02 = -1.64;
        handles. E022 = -1.20;
        handles. tinitial = 0.0012;
        handles.trev1 = 3.5712;
        handles.trev2 = 9.5616;
        handles.trev3=15.6096;
        handles.delta = 0.08;
    elseif handles.W==5 && 0.25<handles.nu && handles.nu<0.4
        handles.nu1=0.25;
        handles.Do1_1 = 0.45E-05;
        handles.DR1_1 = 2.4E-05;
        handles.E01_1 = -0.43;
        handles.E012_1 = -0.25;
        handles.Do2_1 = 1.55E-05;
        handles.DR2_1 = 1.35E-04;
        handles.E02_1 = -1.65;
        handles.E022_1 = -1.2;
        handles.tinitial_1 = 0.0012;
        handles.trev1_1 = 3.5712;
        handles.trev2_1 = 9.5616;
        handles.trev3_1=15.6096;
        handles.delta_1 = 0.08;

        handles.nu2=0.4;
        handles.Do1_2 = 0.45E-05;
        handles.DR1_2 = 2.4E-05;
        handles.E01_2 = -0.43;
        handles.E012_2 = -0.25;
        handles.Do2_2 = 1.45E-05;
        handles.DR2_2 = 0.85E-04;
        handles.E02_2 = -1.67;
        handles.E022_2 = -1.19;
        handles.tinitial_2 = 0.006;
```

```
        handles.trev1_2 = 2.4840;
        handles.trev2_2 = 6.0120;
        handles.delta_2 = 0.08;

    handles.Do1=handles.Do1_1+(((handles.nu-handles.nu1)*(handles.Do1_2-
handles.Do1_1))/(handles.nu2-handles.nu1));
    handles.DR1=handles.DR1_1+(((handles.nu-handles.nu1)*(handles.DR1_2-
handles.DR1_1))/(handles.nu2-handles.nu1));
    handles.E01=handles.E01_1+(((handles.nu-handles.nu1)*(handles.E01_2-
handles.E01_1))/(handles.nu2-handles.nu1));
    handles.E012=handles.E012_1+(((handles.nu-handles.nu1)*(handles.E012_2-
handles.E012_1))/(handles.nu2-handles.nu1));

    handles.Do2=handles.Do2_1+(((handles.nu-handles.nu1)*(handles.Do2_2-
handles.Do2_1))/(handles.nu2-handles.nu1));
    handles.DR2=handles.DR2_1+(((handles.nu-handles.nu1)*(handles.DR2_2-
handles.DR2_1))/(handles.nu2-handles.nu1));
    handles.E02=handles.E02_1+(((handles.nu-handles.nu1)*(handles.E02_2-
handles.E02_1))/(handles.nu2-handles.nu1));
    handles.E022=handles.E022_1+(((handles.nu-handles.nu1)*(handles.E022_2-
handles.E022_1))/(handles.nu2-handles.nu1));

    handles.trev1=handles.trev1_1+(((handles.nu-
handles.nu1)*(handles.trev1_2-handles.trev1_1))/(handles.nu2-handles.nu1));
    handles.trev2=handles.trev2_1+(((handles.nu-
handles.nu1)*(handles.trev2_2-handles.trev2_1))/(handles.nu2-handles.nu1));
    handles.tinitial=handles.tinitial_1+(((handles.nu-
handles.nu1)*(handles.tinitial_2-handles.tinitial_1))/(handles.nu2-
handles.nu1));
    handles.delta=handles.delta_1+(((handles.nu-
handles.nu1)*(handles.delta_2-handles.delta_1))/(handles.nu2-handles.nu1));
    elseif handles.W == 5 && handles.nu == 0.4
        handles.Do1 = 0.45E-05;
        handles.DR1 = 2.4E-05;
        handles.E01 = -0.43;
        handles.E012 = -0.25;
        handles.Do2 = 1.45E-05;
        handles.DR2 = 0.85E-04;
        handles.E02 = -1.67;
        handles.E022 = -1.19;
        handles.tinitial = 0.006;
        handles.trev1 = 2.4840;
        handles.trev2 = 6.0120;
        handles.delta = 0.08;
    elseif handles.W==5 && 0.4<handles.nu && handles.nu<0.6
        handles.nu1=0.4;
        handles.Do1_1 = 0.45E-05;
        handles.DR1_1 = 2.4E-05;
        handles.E01_1 = -0.43;
        handles.E012_1 = -0.25;
        handles.Do2_1 = 1.45E-05;
        handles.DR2_1 = 0.85E-04;
        handles.E02_1 = -1.67;
        handles.E022_1 = -1.19;
        handles.tinitial_1 = 0.006;
        handles.trev1_1 = 2.4840;
```

```
        handles.trev2_1 = 6.0120;
        handles.delta_1 = 0.08;

        handles.nu2=0.6;
        handles.Do1_2 = 0.45E-05;
        handles.DR1_2 = 2.4E-05;
        handles.E01_2 = -0.43;
        handles.E012_2 = -0.25;
        handles.Do2_2 = 1.5E-05;
        handles.DR2_2 = 0.7E-04;
        handles.E02_2 = -1.67;
        handles.E022_2 = -1.16;
        handles.tinitial_2 = 0.006;
        handles.trev1_2 = 1.636;
        handles.trev2_2 = 3.976;
        handles.delta_2 = 0.04;

    handles.Do1=handles.Do1_1+(((handles.nu-handles.nu1)*(handles.Do1_2-
handles.Do1_1))/(handles.nu2-handles.nu1));
    handles.DR1=handles.DR1_1+(((handles.nu-handles.nu1)*(handles.DR1_2-
handles.DR1_1))/(handles.nu2-handles.nu1));
    handles.E01=handles.E01_1+(((handles.nu-handles.nu1)*(handles.E01_2-
handles.E01_1))/(handles.nu2-handles.nu1));
    handles.E012=handles.E012_1+(((handles.nu-handles.nu1)*(handles.E012_2-
handles.E012_1))/(handles.nu2-handles.nu1));

    handles.Do2=handles.Do2_1+(((handles.nu-handles.nu1)*(handles.Do2_2-
handles.Do2_1))/(handles.nu2-handles.nu1));
    handles.DR2=handles.DR2_1+(((handles.nu-handles.nu1)*(handles.DR2_2-
handles.DR2_1))/(handles.nu2-handles.nu1));
    handles.E02=handles.E02_1+(((handles.nu-handles.nu1)*(handles.E02_2-
handles.E02_1))/(handles.nu2-handles.nu1));
    handles.E022=handles.E022_1+(((handles.nu-handles.nu1)*(handles.E022_2-
handles.E022_1))/(handles.nu2-handles.nu1));

    handles.trev1=handles.trev1_1+(((handles.nu-
handles.nu1)*(handles.trev1_2-handles.trev1_1))/(handles.nu2-handles.nu1));
    handles.trev2=handles.trev2_1+(((handles.nu-
handles.nu1)*(handles.trev2_2-handles.trev2_1))/(handles.nu2-handles.nu1));
    handles.tinitial=handles.tinitial_1+(((handles.nu-
handles.nu1)*(handles.tinitial_2-handles.tinitial_1))/(handles.nu2-
handles.nu1));
    handles.delta=handles.delta_1+(((handles.nu-
handles.nu1)*(handles.delta_2-handles.delta_1))/(handles.nu2-handles.nu1));
    elseif handles.W == 5 && handles.nu == 0.6
        handles.Do1 = 0.45E-05;
        handles.DR1 = 2.4E-05;
        handles.E01 = -0.43;
        handles.E012 = -0.25;
        handles.Do2 = 1.5E-05;
        handles.DR2 = 0.7E-04;
        handles.E02 = -1.67;
        handles.E022 = -1.16;
        handles.tinitial = 0.006;
        handles.trev1 = 1.636;
        handles.trev2 = 3.976;
```

```
        handles.delta = 0.04;
    elseif handles.W==5 && 0.6<handles.nu && handles.nu<0.9
        handles.nu1=0.6;
        handles.Do1_1 = 0.45E-05;
        handles.DR1_1 = 2.4E-05;
        handles.E01_1 = -0.43;
        handles.E012_1 = -0.25;
        handles.Do2_1 = 1.5E-05;
        handles.DR2_1 = 0.7E-04;
        handles.E02_1 = -1.67;
        handles.E022_1 = -1.16;
        handles.tinitial_1 = 0.006;
        handles.trev1_1 = 1.636;
        handles.trev2_1 = 3.976;
        handles.delta_1 = 0.04;

        handles.nu2=0.9;
        handles.Do1_2 = 4.5e-06;
        handles.DR1_2 = 2.4e-05;
        handles.E01_2 = -0.43;
        handles.E012_2 = -0.23;
        handles.Do2_2 = 1.18e-05;
        handles.DR2_2 = 5.3e-05;
        handles.E02_2 = -1.7;
        handles.E022_2 = -1.13;
        handles.tinitial_2 = 0.004;
        handles.trev1_2 = 0.9575;
        handles.trev2_2 = 2.66167;
        handles.delta_2 = 0.04;

    handles.Do1=handles.Do1_1+(((handles.nu-handles.nu1)*(handles.Do1_2-
handles.Do1_1))/(handles.nu2-handles.nu1));
    handles.DR1=handles.DR1_1+(((handles.nu-handles.nu1)*(handles.DR1_2-
handles.DR1_1))/(handles.nu2-handles.nu1));
    handles.E01=handles.E01_1+(((handles.nu-handles.nu1)*(handles.E01_2-
handles.E01_1))/(handles.nu2-handles.nu1));
    handles.E012=handles.E012_1+(((handles.nu-handles.nu1)*(handles.E012_2-
handles.E012_1))/(handles.nu2-handles.nu1));

    handles.Do2=handles.Do2_1+(((handles.nu-handles.nu1)*(handles.Do2_2-
handles.Do2_1))/(handles.nu2-handles.nu1));
    handles.DR2=handles.DR2_1+(((handles.nu-handles.nu1)*(handles.DR2_2-
handles.DR2_1))/(handles.nu2-handles.nu1));
    handles.E02=handles.E02_1+(((handles.nu-handles.nu1)*(handles.E02_2-
handles.E02_1))/(handles.nu2-handles.nu1));
    handles.E022=handles.E022_1+(((handles.nu-handles.nu1)*(handles.E022_2-
handles.E022_1))/(handles.nu2-handles.nu1));

    handles.trev1=handles.trev1_1+(((handles.nu-
handles.nu1)*(handles.trev1_2-handles.trev1_1))/(handles.nu2-handles.nu1));
    handles.trev2=handles.trev2_1+(((handles.nu-
handles.nu1)*(handles.trev2_2-handles.trev2_1))/(handles.nu2-handles.nu1));
    handles.tinitial=handles.tinitial_1+(((handles.nu-
handles.nu1)*(handles.tinitial_2-handles.tinitial_1))/(handles.nu2-
handles.nu1));
```

```
    handles.delta=handles.delta_1+(((handles.nu-
handles.nu1)*(handles.delta_2-handles.delta_1))/(handles.nu2-handles.nu1));
    elseif handles.W == 5 && handles.nu == 0.9
        handles.Do1 =4.5e-06;
        handles.DR1 = 2.4e-05;
        handles.E01 = -0.43;
        handles.E012 = -0.23;
        handles.Do2 = 1.18e-05;
        handles.DR2 = 5.3e-05;
        handles.E02 = -1.7;
        handles.E022 =-1.13;
        handles.tinitial = 0.004;
        handles.trev1 = 0.9575;
        handles.trev2 = 2.66167;
        handles.delta = 0.04;
    elseif handles.W == 5 && handles.nu == 1
        handles.Do1 =4.5e-06;
        handles.DR1 = 2.4e-05;
        handles.E01 = -0.43;
        handles.E012 = -0.23;
        handles.Do2 = 1.18e-05;
        handles.DR2 = 5.2e-05;
        handles.E02 = -1.7;
        handles.E022 =-1.13;
        handles.tinitial = 0.004;
        handles.trev1 = 0.8136;
        handles.trev2 = 2.3952;
        handles.delta = 0.04;
    elseif handles.W==5 && 1<handles.nu && handles.nu<2
        handles.nu1=1;
        handles.Do1_1 = 4.5e-06;
        handles.DR1_1 = 2.4E-05;
        handles.E01_1 = -0.43;
        handles.E012_1 = -0.23;
        handles.Do2_1 = 1.18E-05;
        handles.DR2_1 = 5.2e-05;
        handles.E02_1 = -1.7;
        handles.E022_1 = -1.13;
        handles.tinitial_1 = 0.004;
        handles.trev1_1 = 0.8136;
        handles.trev2_1 = 2.3952;
        handles.delta_1 = 0.04;

        handles.nu2=2;
        handles.Do1_2 = 4.7e-06;
        handles.DR1_2 = 2.4e-05;
        handles.E01_2 = -0.49;
        handles.E012_2 = -0.2;
        handles.Do2_2 = 1.14e-05;
        handles.DR2_2 = 4.7e-05;
        handles.E02_2 = -1.78;
        handles.E022_2 = -1;
        handles.tinitial_2 = 0.004;
        handles.trev1_2 = 0.3696;
        handles.trev2_2 = 1.1988;
        handles.delta_2 = 0.02;
```

```
    handles.Do1=handles.Do1_1+(((handles.nu-handles.nu1)*(handles.Do1_2-
handles.Do1_1))/(handles.nu2-handles.nu1));
    handles.DR1=handles.DR1_1+(((handles.nu-handles.nu1)*(handles.DR1_2-
handles.DR1_1))/(handles.nu2-handles.nu1));
    handles.E01=handles.E01_1+(((handles.nu-handles.nu1)*(handles.E01_2-
handles.E01_1))/(handles.nu2-handles.nu1));
    handles.E012=handles.E012_1+(((handles.nu-handles.nu1)*(handles.E012_2-
handles.E012_1))/(handles.nu2-handles.nu1));

    handles.Do2=handles.Do2_1+(((handles.nu-handles.nu1)*(handles.Do2_2-
handles.Do2_1))/(handles.nu2-handles.nu1));
    handles.DR2=handles.DR2_1+(((handles.nu-handles.nu1)*(handles.DR2_2-
handles.DR2_1))/(handles.nu2-handles.nu1));
    handles.E02=handles.E02_1+(((handles.nu-handles.nu1)*(handles.E02_2-
handles.E02_1))/(handles.nu2-handles.nu1));
    handles.E022=handles.E022_1+(((handles.nu-handles.nu1)*(handles.E022_2-
handles.E022_1))/(handles.nu2-handles.nu1));

    handles.trev1=handles.trev1_1+(((handles.nu-
handles.nu1)*(handles.trev1_2-handles.trev1_1))/(handles.nu2-handles.nu1));
    handles.trev2=handles.trev2_1+(((handles.nu-
handles.nu1)*(handles.trev2_2-handles.trev2_1))/(handles.nu2-handles.nu1));
    handles.tinitial=handles.tinitial_1+(((handles.nu-
handles.nu1)*(handles.tinitial_2-handles.tinitial_1))/(handles.nu2-
handles.nu1));
    handles.delta=handles.delta_1+(((handles.nu-
handles.nu1)*(handles.delta_2-handles.delta_1))/(handles.nu2-handles.nu1));
        elseif handles.W == 5 && handles.nu == 2
        handles.nu2=2;
        handles.Do1 = 4.7e-06;
        handles.DR1 = 2.4e-05;
        handles.E01 = -0.49;
        handles.E012 = -0.2;
        handles.Do2 = 1.14e-05;
        handles.DR2 = 4.7e-05;
        handles.E02 = -1.78;
        handles.E022 = -1;
        handles.tinitial = 0.004;
        handles.trev1 = 0.3696;
        handles.trev2 = 1.1988;
        handles.delta = 0.02;
    elseif handles.W == 7.5 && handles.nu == 0.4
        handles.Do1 = 0.45E-05;
        handles.DR1 = 1.85E-05;
        handles.E01 = -0.430;
        handles.E012 = -0.2;
        handles.Do2 = 1.4E-05;
        handles.DR2 = 0.95E-04;
        handles.E02 = -1.7;
        handles.E022 = -1.09;
        handles.tinitial = 0.0012;
        handles.trev1 = 1.92;
        handles.trev2 = 5.988;
        handles.delta = 0.03;
    elseif handles.W==7.5 && 0.4<handles.nu && handles.nu<0.5
```

```
        handles.nu1=0.4;
        handles.Do1_1 = 0.45E-05;
        handles.DR1_1 = 1.85E-05;
        handles.E01_1 = -0.430;
        handles.E012_1 = -0.2;
        handles.Do2_1 = 1.4E-05;
        handles.DR2_1 = 0.95E-04;
        handles.E02_1 = -1.7;
        handles.E022_1 = -1.09;
        handles.tinitial_1 = 0.0012;
        handles.trev1_1 = 1.92;
        handles.trev2_1 = 5.988;
        handles.delta_1 = 0.03;

        handles.nu2=0.5;
        handles.Do1_2 = 0.49E-05;
        handles.DR1_2 = 1.85E-05;
        handles.E01_2 = -0.430;
        handles.E012_2 = -0.2;
        handles.Do2_2 = 1.4E-05;
        handles.DR2_2 = 0.75E-04;
        handles.E02_2 = -1.7;
        handles.E022_2 = -1.07;
        handles.tinitial_2 = 0.0012;
        handles.trev1_2 = 1.3248;
        handles.trev2_2 = 4.7404;
        handles.delta_2 = 0.03;
    handles.Do1=handles.Do1_1+(((handles.nu-handles.nu1)*(handles.Do1_2-
handles.Do1_1))/(handles.nu2-handles.nu1));
    handles.DR1=handles.DR1_1+(((handles.nu-handles.nu1)*(handles.DR1_2-
handles.DR1_1))/(handles.nu2-handles.nu1));
    handles.E01=handles.E01_1+(((handles.nu-handles.nu1)*(handles.E01_2-
handles.E01_1))/(handles.nu2-handles.nu1));
    handles.E012=handles.E012_1+(((handles.nu-handles.nu1)*(handles.E012_2-
handles.E012_1))/(handles.nu2-handles.nu1));

    handles.Do2=handles.Do2_1+(((handles.nu-handles.nu1)*(handles.Do2_2-
handles.Do2_1))/(handles.nu2-handles.nu1));
    handles.DR2=handles.DR2_1+(((handles.nu-handles.nu1)*(handles.DR2_2-
handles.DR2_1))/(handles.nu2-handles.nu1));
    handles.E02=handles.E02_1+(((handles.nu-handles.nu1)*(handles.E02_2-
handles.E02_1))/(handles.nu2-handles.nu1));
    handles.E022=handles.E022_1+(((handles.nu-handles.nu1)*(handles.E022_2-
handles.E022_1))/(handles.nu2-handles.nu1));

    handles.trev1=handles.trev1_1+(((handles.nu-
handles.nu1)*(handles.trev1_2-handles.trev1_1))/(handles.nu2-handles.nu1));
    handles.trev2=handles.trev2_1+(((handles.nu-
handles.nu1)*(handles.trev2_2-handles.trev2_1))/(handles.nu2-handles.nu1));
    handles.tinitial=handles.tinitial_1+(((handles.nu-
handles.nu1)*(handles.tinitial_2-handles.tinitial_1))/(handles.nu2-
handles.nu1));
    handles.delta=handles.delta_1+(((handles.nu-
handles.nu1)*(handles.delta_2-handles.delta_1))/(handles.nu2-handles.nu1));
    elseif handles.W == 7.5 && handles.nu == 0.5
        handles.Do1 = 0.49E-05;
```

```
        handles.DR1 = 1.85E-05;
        handles.E01 = -0.430;
        handles.E012 = -0.2;
        handles.Do2 = 1.4E-05;
        handles.DR2 = 0.75E-04;
        handles.E02 = -1.7;
        handles.E022 = -1.07;
        handles.tinitial = 0.0012;
        handles.trev1 = 1.3248;
        handles.trev2 = 4.7404;
        handles.delta = 0.03;
    elseif handles.W==7.5 && 0.5<handles.nu && handles.nu<1
        handles.nu1=0.5;
        handles.Do1_1 = 0.45E-05;
        handles.DR1_1 = 1.85E-05;
        handles.E01_1 = -0.430;
        handles.E012_1 = -0.2;
        handles.Do2_1 = 0.95E-04;
        handles.DR2_1 = 1.4E-05;
        handles.E02_1 = -1.7;
        handles.E022_1 = -1.09;
        handles.tinitial_1 = 0.0012;
        handles.trev1_1 = 1.656;
        handles.trev2_1 = 5.994;
        handles.delta_1 = 0.03;

        handles.nu2=1;
        handles.Do1_2 = 0.5E-05;
        handles.DR1_2 = 2.2E-05;
        handles.E01_2 = -0.5;
        handles.E012_2 = -0.16;
        handles.Do2_2 = 1.05E-05;
        handles.DR2_2 = 0.50E-04;
        handles.E02_2 = -1.75;
        handles.E022_2 = -0.98;
        handles.tinitial_2 = 0.0012;
        handles.trev1_2 = 0.6528;
        handles.trev2_2 = 2.3928;
        handles.delta_2 = 0.02;
    handles.Do1=handles.Do1_1+(((handles.nu-handles.nu1)*(handles.Do1_2-
handles.Do1_1))/(handles.nu2-handles.nu1));
    handles.DR1=handles.DR1_1+(((handles.nu-handles.nu1)*(handles.DR1_2-
handles.DR1_1))/(handles.nu2-handles.nu1));
    handles.E01=handles.E01_1+(((handles.nu-handles.nu1)*(handles.E01_2-
handles.E01_1))/(handles.nu2-handles.nu1));
    handles.E012=handles.E012_1+(((handles.nu-handles.nu1)*(handles.E012_2-
handles.E012_1))/(handles.nu2-handles.nu1));

    handles.Do2=handles.Do2_1+(((handles.nu-handles.nu1)*(handles.Do2_2-
handles.Do2_1))/(handles.nu2-handles.nu1));
    handles.DR2=handles.DR2_1+(((handles.nu-handles.nu1)*(handles.DR2_2-
handles.DR2_1))/(handles.nu2-handles.nu1));
    handles.E02=handles.E02_1+(((handles.nu-handles.nu1)*(handles.E02_2-
handles.E02_1))/(handles.nu2-handles.nu1));
    handles.E022=handles.E022_1+(((handles.nu-handles.nu1)*(handles.E022_2-
handles.E022_1))/(handles.nu2-handles.nu1));
```

```
    handles.trev1=handles.trev1_1+(((handles.nu-
handles.nu1)*(handles.trev1_2-handles.trev1_1))/(handles.nu2-handles.nu1));
    handles.trev2=handles.trev2_1+(((handles.nu-
handles.nu1)*(handles.trev2_2-handles.trev2_1))/(handles.nu2-handles.nu1));
    handles.tinitial=handles.tinitial_1+(((handles.nu-
handles.nu1)*(handles.tinitial_2-handles.tinitial_1))/(handles.nu2-
handles.nu1));
    handles.delta=handles.delta_1+(((handles.nu-
handles.nu1)*(handles.delta_2-handles.delta_1))/(handles.nu2-handles.nu1));
    elseif handles.W==7.5 && handles.nu==1
        handles.nu2=1;
        handles.Do1 = 0.5E-05;
        handles.DR1 = 2.2E-05;
        handles.E01 = -0.5;
        handles.E012 = -0.16;
        handles.Do2 = 1.05E-05;
        handles.DR2 = 0.50E-04;
        handles.E02 = -1.75;
        handles.E022 = -0.98;
        handles.tinitial = 0.0012;
        handles.trev1 = 0.6528;
        handles.trev2 = 2.3928;
        handles.delta = 0.02;
        elseif handles.W==7.5 && 1<handles.nu && handles.nu<1.4
        handles.nu2=1;
        handles.Do1_1 = 0.5E-05;
        handles.DR1_1 = 2.2E-05;
        handles.E01_1 = -0.5;
        handles.E012_1 = -0.16;
        handles.Do2_1 = 1.05E-05;
        handles.DR2_1 = 0.50E-04;
        handles.E02_1 = -1.75;
        handles.E022_1 = -0.98;
        handles.tinitial_1 = 0.0012;
        handles.trev1_1 = 0.6528;
        handles.trev2_1 = 2.3928;
        handles.delta_1 = 0.02;

        handles.nu1=1.4;
        handles.Do1_2 = 0.5E-05;
        handles.DR1_2 = 2.2E-05;
        handles.E01_2 = -0.5;
        handles.E012_2 = -0.16;
        handles.Do2_2 = 0.99E-05;
        handles.DR2_2 = 0.418E-04;
        handles.E02_2 = -1.79;
        handles.E022_2 = -0.94;
        handles.tinitial_2 = 0.0012;
        handles.trev1_2 = 0.4936;
        handles.trev2_2 = 1.7243;
        handles.delta_2 = 0.02;

    handles.Do1=handles.Do1_1+(((handles.nu-handles.nu1)*(handles.Do1_2-
handles.Do1_1))/(handles.nu2-handles.nu1));
```

```
    handles.DR1=handles.DR1_1+(((handles.nu-handles.nu1)*(handles.DR1_2-
handles.DR1_1))/(handles.nu2-handles.nu1));
    handles.E01=handles.E01_1+(((handles.nu-handles.nu1)*(handles.E01_2-
handles.E01_1))/(handles.nu2-handles.nu1));
    handles.E012=handles.E012_1+(((handles.nu-handles.nu1)*(handles.E012_2-
handles.E012_1))/(handles.nu2-handles.nu1));

    handles.Do2=handles.Do2_1+(((handles.nu-handles.nu1)*(handles.Do2_2-
handles.Do2_1))/(handles.nu2-handles.nu1));
    handles.DR2=handles.DR2_1+(((handles.nu-handles.nu1)*(handles.DR2_2-
handles.DR2_1))/(handles.nu2-handles.nu1));
    handles.E02=handles.E02_1+(((handles.nu-handles.nu1)*(handles.E02_2-
handles.E02_1))/(handles.nu2-handles.nu1));
    handles.E022=handles.E022_1+(((handles.nu-handles.nu1)*(handles.E022_2-
handles.E022_1))/(handles.nu2-handles.nu1));

    handles.trev1=handles.trev1_1+(((handles.nu-
handles.nu1)*(handles.trev1_2-handles.trev1_1))/(handles.nu2-handles.nu1));
    handles.trev2=handles.trev2_1+(((handles.nu-
handles.nu1)*(handles.trev2_2-handles.trev2_1))/(handles.nu2-handles.nu1));
    handles.tinitial=handles.tinitial_1+(((handles.nu-
handles.nu1)*(handles.tinitial_2-handles.tinitial_1))/(handles.nu2-
handles.nu1));
    handles.delta=handles.delta_1+(((handles.nu-
handles.nu1)*(handles.delta_2-handles.delta_1))/(handles.nu2-handles.nu1));
    elseif handles.W==7.5 && handles.nu==1.4
        handles.nu1=1.4;
        handles.Do1 = 0.5E-05;
        handles.DR1 = 2.2E-05;
        handles.E01 = -0.5;
        handles.E012 = -0.16;
        handles.Do2 = 0.99E-05;
        handles.DR2 = 0.418E-04;
        handles.E02 = -1.79;
        handles.E022 = -0.94;
        handles.tinitial = 0.0012;
        handles.trev1 = 0.4936;
        handles.trev2 = 1.7243;
        handles.delta = 0.02;
    elseif handles.W==7.5 && 1.4<handles.nu && handles.nu<2
        handles.nu1=1.4;
        handles.Do1_1 = 0.5E-05;
        handles.DR1_1 = 2.2E-05;
        handles.E01_1 = -0.5;
        handles.E012_1 = -0.16;
        handles.Do2_1 = 0.99E-05;
        handles.DR2_1 = 0.418E-04;
        handles.E02_1 = -1.79;
        handles.E022_1 = -0.94;
        handles.tinitial_1 = 0.0012;
        handles.trev1_1 = 0.4936;
        handles.trev2_1 = 1.7243;
        handles.delta_1 = 0.02;


        handles.nu2=2;
```

```
        handles.Do1_1 = 0.5E-05;
        handles.DR1_1 = 2.45E-05;
        handles.E01_1 = -0.5;
        handles.E012_1 = -0.145;
        handles.Do2_1 = 0.94E-05;
        handles.DR2_1 = 0.34E-04;
        handles.E02_1 = -1.81;
        handles.E022_1 = -0.88;
        handles.tinitial_1 = 0.0012;
        handles.trev1_1 = 0.3396;
        handles.trev2_1 = 1.1952
        handles.delta_1 = 0.02;
    handles.Do1=handles.Do1_1+(((handles.nu-handles.nu1)*(handles.Do1_2-
handles.Do1_1))/(handles.nu2-handles.nu1));
    handles.DR1=handles.DR1_1+(((handles.nu-handles.nu1)*(handles.DR1_2-
handles.DR1_1))/(handles.nu2-handles.nu1));
    handles.E01=handles.E01_1+(((handles.nu-handles.nu1)*(handles.E01_2-
handles.E01_1))/(handles.nu2-handles.nu1));
    handles.E012=handles.E012_1+(((handles.nu-handles.nu1)*(handles.E012_2-
handles.E012_1))/(handles.nu2-handles.nu1));

    handles.Do2=handles.Do2_1+(((handles.nu-handles.nu1)*(handles.Do2_2-
handles.Do2_1))/(handles.nu2-handles.nu1));
    handles.DR2=handles.DR2_1+(((handles.nu-handles.nu1)*(handles.DR2_2-
handles.DR2_1))/(handles.nu2-handles.nu1));
    handles.E02=handles.E02_1+(((handles.nu-handles.nu1)*(handles.E02_2-
handles.E02_1))/(handles.nu2-handles.nu1));
    handles.E022=handles.E022_1+(((handles.nu-handles.nu1)*(handles.E022_2-
handles.E022_1))/(handles.nu2-handles.nu1));

    handles.trev1=handles.trev1_1+(((handles.nu-
handles.nu1)*(handles.trev1_2-handles.trev1_1))/(handles.nu2-handles.nu1));
    handles.trev2=handles.trev2_1+(((handles.nu-
handles.nu1)*(handles.trev2_2-handles.trev2_1))/(handles.nu2-handles.nu1));
    handles.tinitial=handles.tinitial_1+(((handles.nu-
handles.nu1)*(handles.tinitial_2-handles.tinitial_1))/(handles.nu2-
handles.nu1));
    handles.delta=handles.delta_1+(((handles.nu-
handles.nu1)*(handles.delta_2-handles.delta_1))/(handles.nu2-handles.nu1));
    elseif handles.W==7.5 && handles.nu==2
        handles.nu1=2;
        handles.Do1 = 0.5E-05;
        handles.DR1 = 2.45E-05;
        handles.E01 = -0.5;
        handles.E012 = -0.145;
        handles.Do2 = 0.94E-05;
        handles.DR2 = 0.34E-04;
        handles.E02 = -1.81;
        handles.E022 = -0.88;
        handles.tinitial = 0.0012;
        handles.trev1 = 0.3396;
        handles.trev2 = 1.1952;
        handles.delta = 0.02;
    elseif handles.W==10 && handles.nu==0.2
        handles.Do1=0.25E-05;
        handles.DR1=0.75E-05;
```

```
        handles.E01=-0.4;
        handles.E012=-0.18;
        handles.Do2=0.93E-05;
        handles.DR2=0.9E-04;
        handles.E02=-1.68;
        handles.E022=-1.05;
        handles.tinitial = 0.0012;
        handles.trev1=3.652;
        handles.trev2=11.98;
        handles.delta = 0.08;
    elseif handles.W==10 && 0.2<handles.nu && handles.nu<0.5
        handles.nu1=0.2;
        handles.Do1_1=0.25E-05;
        handles.DR1_1=0.75E-05;
        handles.E01_1=-0.4;
        handles.E012_1=-0.18;
        handles.Do2_1=0.93E-05;
        handles.DR2_1=0.9E-04;
        handles.E02_1=-1.68;
        handles.E022_1=-1.05;
        handles.tinitial_1 = 0.0012;
        handles.trev1_1=3.652;
        handles.trev2_1=11.98;
        handles.delta_1 = 0.08;


        handles.nu2=0.5;
        handles.Do1_2=0.2E-05;
        handles.DR1_2=1.5E-05;
        handles.E01_2=-0.55;
        handles.E012_2=-0.18;
        handles.Do2_2=0.83E-05;
        handles.DR2_2=0.44E-04;
        handles.E02_2=-1.73;
        handles.E022_2=-1;
        handles.tinitial_2 = 0.0012;
        handles.trev1_2=1.2832;
        handles.trev2_2=4.792;
        handles.delta_2 = 0.08;
    handles.Do1=handles.Do1_1+(((handles.nu-handles.nu1)*(handles.Do1_2-
handles.Do1_1))/(handles.nu2-handles.nu1));
    handles.DR1=handles.DR1_1+(((handles.nu-handles.nu1)*(handles.DR1_2-
handles.DR1_1))/(handles.nu2-handles.nu1));
    handles.E01=handles.E01_1+(((handles.nu-handles.nu1)*(handles.E01_2-
handles.E01_1))/(handles.nu2-handles.nu1));
    handles.E012=handles.E012_1+(((handles.nu-handles.nu1)*(handles.E012_2-
handles.E012_1))/(handles.nu2-handles.nu1));

    handles.Do2=handles.Do2_1+(((handles.nu-handles.nu1)*(handles.Do2_2-
handles.Do2_1))/(handles.nu2-handles.nu1));
    handles.DR2=handles.DR2_1+(((handles.nu-handles.nu1)*(handles.DR2_2-
handles.DR2_1))/(handles.nu2-handles.nu1));
    handles.E02=handles.E02_1+(((handles.nu-handles.nu1)*(handles.E02_2-
handles.E02_1))/(handles.nu2-handles.nu1));
    handles.E022=handles.E022_1+(((handles.nu-handles.nu1)*(handles.E022_2-
handles.E022_1))/(handles.nu2-handles.nu1));
```

```
    handles.trev1=handles.trev1_1+(((handles.nu-
handles.nu1)*(handles.trev1_2-handles.trev1_1))/(handles.nu2-handles.nu1));
    handles.trev2=handles.trev2_1+(((handles.nu-
handles.nu1)*(handles.trev2_2-handles.trev2_1))/(handles.nu2-handles.nu1));
    handles.tinitial=handles.tinitial_1+(((handles.nu-
handles.nu1)*(handles.tinitial_2-handles.tinitial_1))/(handles.nu2-
handles.nu1));
    handles.delta=handles.delta_1+(((handles.nu-
handles.nu1)*(handles.delta_2-handles.delta_1))/(handles.nu2-handles.nu1));
    elseif handles.W==10 && handles.nu==0.5
        handles.Do1=0.2E-05;
        handles.DR1=1.5E-05;
        handles.E01=-0.55;
        handles.E012=-0.18;
        handles.Do2=0.83E-05;
        handles.DR2=0.44E-04;
        handles.E02=-1.73;
        handles.E022=-1;
        handles.tinitial = 0.0012;
        handles.trev1=1.2832;
        handles.trev2=4.792;
        handles.delta = 0.05;
        elseif handles.W==10 && 0.5<handles.nu && handles.nu<0.8
        handles.nu1=0.5;
        handles.Do1_1=0.2E-05;
        handles.DR1_1=1.5E-05;
        handles.E01_1=-0.55;
        handles.E012_1=-0.18;
        handles.Do2_1=0.83E-05;
        handles.DR2_1=0.44E-04;
        handles.E02_1=-1.73;
        handles.E022_1=-1;
        handles.tinitial_1 = 0.0012;
        handles.trev1_1=1.2832;
        handles.trev2_1=4.792;
        handles.delta_1 = 0.05;

        handles.nu2=0.8;
        handles.Do1_2=0.13E-05;
        handles.DR1_2=1.8E-05;
        handles.E01_2=-0.55;
        handles.E012_2=-0.18;
        handles.Do2_2=0.65E-05;
        handles.DR2_2=0.3E-04;
        handles.E02_2=-1.75;
        handles.E022_2=-0.95;
        handles.tinitial_2=0.0012;
        handles.trev1_2=0.8460;
        handles.trev2_2=2.9910;
        handles.delta_2 = 0.05;


    handles.Do1=handles.Do1_1+(((handles.nu-handles.nu1)*(handles.Do1_2-
handles.Do1_1))/(handles.nu2-handles.nu1));
```

```
    handles.DR1=handles.DR1_1+(((handles.nu-handles.nu1)*(handles.DR1_2-
handles.DR1_1))/(handles.nu2-handles.nu1));
    handles.E01=handles.E01_1+(((handles.nu-handles.nu1)*(handles.E01_2-
handles.E01_1))/(handles.nu2-handles.nu1));
    handles.E012=handles.E012_1+(((handles.nu-handles.nu1)*(handles.E012_2-
handles.E012_1))/(handles.nu2-handles.nu1));

    handles.Do2=handles.Do2_1+(((handles.nu-handles.nu1)*(handles.Do2_2-
handles.Do2_1))/(handles.nu2-handles.nu1));
    handles.DR2=handles.DR2_1+(((handles.nu-handles.nu1)*(handles.DR2_2-
handles.DR2_1))/(handles.nu2-handles.nu1));
    handles.E02=handles.E02_1+(((handles.nu-handles.nu1)*(handles.E02_2-
handles.E02_1))/(handles.nu2-handles.nu1));
    handles.E022=handles.E022_1+(((handles.nu-handles.nu1)*(handles.E022_2-
handles.E022_1))/(handles.nu2-handles.nu1));

    handles.trev1=handles.trev1_1+(((handles.nu-
handles.nu1)*(handles.trev1_2-handles.trev1_1))/(handles.nu2-handles.nu1));
    handles.trev2=handles.trev2_1+(((handles.nu-
handles.nu1)*(handles.trev2_2-handles.trev2_1))/(handles.nu2-handles.nu1));
    handles.tinitial=handles.tinitial_1+(((handles.nu-
handles.nu1)*(handles.tinitial_2-handles.tinitial_1))/(handles.nu2-
handles.nu1));
    handles.delta=handles.delta_1+(((handles.nu-
handles.nu1)*(handles.delta_2-handles.delta_1))/(handles.nu2-handles.nu1));
    elseif handles.W==10 && handles.nu==0.8
        handles.Do1=0.13E-05;
        handles.DR1=1.8E-05;
        handles.E01=-0.55;
        handles.E012=-0.18;
        handles.Do2=0.65E-05;
        handles.DR2=0.3E-04;
        handles.E02=-1.75;
        handles.E022=-0.95;
        handles.tinitial=0.0012;
        handles.trev1=0.8460;
        handles.trev2=2.9910;
        handles.delta = 0.05;
    end
    handles.trev3=handles.trev2+(handles.trev2-handles.trev1);
    handles.landa=handles.trev2;
    handles.tfinal=handles.trev3+(handles.trev1-handles.tinitial);
    guidata(hObject,handles)
    %% Calculation of U concentration in mole/cm3
    handles.F=96485;
    handles.T=773.15;
    handles.R=8.314;
    handles.Ei=0;
    handles.ks=0.00026;
    handles.alpha=0.5;
    handles.n1=1;
    handles.n2=3;
    handles.MCl3=344.39; %UCL3
    handles.RumLiCl=1.502; % g/cm3
    handles.RumKCl=1.527;
    handles.tmLiCl=610; % C
```

```
    handles.tmKCl=771; % C
    handles.kLiCl=0.000432; %g/cm3C
    handles.kKCl=0.000583; %g/cm3C
    handles.RuLiCl=handles.RumLiCl-(handles.kLiCl*((handles.T-274.14)-
handles.tmLiCl));
    handles.RuKCl=handles.RumKCl-(handles.kKCl*((handles.T-274.14)-
handles.tmKCl));
    handles.Ru=(0.5*handles.RuKCl)+(0.5*handles.RuLiCl);
    handles.C=(handles.W/100)*handles.Ru/handles.MCl3; %mol/cm3
    handles.u=238.0289;
    handles.cl=35.453;
    %% %% Reversible Cathodic/Anodic Current and Cathodic Potential:
    % O + ne <---> R
    % U(IV) + e <---> U(III)
    % Do= D U(IV)/U(III)
    % DR= D U(III)/U(IV) %%

    handles.t1=handles.tinitial;
    for w=1:round(abs(handles.trev1-handles.t1)/handles.delta);
        handles.t1=handles.t1+handles.delta;
        handles.tetha=exp((handles.n1*handles.F*(handles.Ei-
handles.E01)/(handles.R*handles.T)));
        handles.gama=sqrt(handles.Do1/handles.DR1);
        handles.tetgama=handles.tetha*handles.gama;

M1=round((handles.n1*handles.F*handles.nu*handles.t1)/(handles.R*handles.T*ha
ndles.delta));
        handles.a=(handles.n1*handles.F*handles.nu)/(handles.R*handles.T);
        S1=zeros(M1,M1);
        for i=1:M1;
            j=0;
            B1(i,1)=1/(2*sqrt(handles.delta)*(1+(handles.tetgama*exp(-
handles.delta*(i)))));
            while j<M1;
                j=j+1;
                N=i;
                if j<N;
                    S1(i,j)=sqrt(N-(j-1))-sqrt(N-j);
                elseif N==j
                    S1(i,j)=1;
                else j>N;
                    S1(i,j)=0;
                end
            end
        end
        x1=S1\B1; % or x=inv(S)*B
        Xat1=x1(i,1);

handles.I1rev(1,w)=handles.n1*handles.F*handles.A*handles.C*sqrt(handles.Do1*
handles.a*pi)*Xat1;
        handles.E1rev(1,w)=handles.E01-
(((handles.R*handles.T)/(handles.n1*handles.F))*log(handles.gama))+(((handles
.R*handles.T)/(handles.n1*handles.F))*(log(handles.tetgama)-
(handles.a*handles.t1)));
    end
    handles.E1=handles.E1rev(1,w);
```

```
    handles.t2=handles.tinitial;
    for w=1:round(abs(handles.trev1-handles.t2)/handles.delta);
        handles.t2=handles.t2+handles.delta;
        handles.tetha=exp((handles.n1*handles.F*(handles.Ei-
handles.E012)/(handles.R*handles.T)));
        handles.gama=sqrt(handles.Do1/handles.DR1);
        handles.tetgama=handles.tetha*handles.gama;

M2=round((handles.n1*handles.F*handles.nu*handles.t2)/(handles.R*handles.T*ha
ndles.delta));
        handles.a=(handles.n1*handles.F*handles.nu)/(handles.R*handles.T);
        S2=zeros(M2,M2);
        for i=1:M2;
            j=0;
            B2(i,1)=1/(2*sqrt(handles.delta)*(1+(handles.tetgama*exp(-
handles.delta*(i)))));
            while j<M2;
                j=j+1;
                N=i;
                if j<N;
                    S2(i,j)=sqrt(N-(j-1))-sqrt(N-j);
                elseif N==j
                    S2(i,j)=1;
                else j>N;
                    S2(i,j)=0;
                end
            end
        end
        x2=S2\B2; % or x=inv(S)*B
        Xat2=x2(i,1);

handles.I12rev(1,w)=handles.n1*handles.F*handles.A*handles.C*sqrt(handles.DR1
*handles.a*pi)*Xat2;
        handles.E12rev(1,w)=handles.E012-
(((handles.R*handles.T)/(handles.n1*handles.F))*log(handles.gama))+(((handles
.R*handles.T)/(handles.n1*handles.F))*(log(handles.tetgama)-
(handles.a*handles.t2)));
    end
    handles.E3=handles.E12rev(1,w);
    %% %% Irreversible Cathodic/Anodic Current:
    % O + ne ----> R      U+3 + 3e -----> U
    % Do=DU(III)/U
    % DR=DU/U(III)
    handles.t=handles.trev1;
    for w=1:round(abs(handles.trev2-handles.t)/handles.delta);
        handles.t=handles.t+handles.delta;
        handles.tetha=exp((handles.n2*handles.F*handles.alpha*(handles.Ei-
handles.E02)/(handles.R*handles.T)));
        handles.gama=sqrt(handles.Do2/handles.DR2);
        handles.tetgama=handles.tetha*handles.gama;

M=round((handles.n2*handles.F*handles.nu*handles.alpha*handles.t)/(handles.R*
handles.T*handles.delta));

handles.b=(handles.n2*handles.F*handles.alpha*handles.nu)/(handles.R*handles.
T);
```

```
        handles.k=handles.ks*exp(((-
handles.alpha*handles.n2*handles.F)/(handles.R*handles.T))*(handles.Ei-
handles.E02));
        handles.u=log(sqrt(pi*handles.Do2*handles.b)/handles.k);
        S3=zeros(M,M);
        for i=1:M;
            j=0;
            B3(i,1)=1/(2*sqrt(handles.delta)*(1+(handles.tetgama*exp(-
handles.delta*(i)))));
            while j<M;
                j=j+1;
                N=i;
                if j<N;
                    S3(i,j)=sqrt(N-(j-1))-sqrt(N-j);
                elseif N==j
                    S3(i,j)=1;
                else j>N;
                    S3(i,j)=0;
                end
            end
        end
        x3=S3\B3; % or x=inv(S)*B
        Xbt3=x3(i,1);

handles.I1irrev(1,w)=(handles.n2*handles.F*handles.A*handles.C*sqrt(handles.D
o2*handles.b*pi)*Xbt3);

handles.E1irrev(1,w)=handles.E02+(((handles.R*handles.T)/(handles.F*handles.n
2*handles.alpha))*(handles.u-(handles.b*handles.t)))-
(((handles.R*handles.T)/(handles.F*handles.n2*handles.alpha))*log(sqrt(pi*han
dles.Do2*handles.b)/handles.ks));
    end
    handles.t=handles.trev1;
    for w=1:round(abs(handles.trev2-handles.t)/(2*handles.delta));
        handles.t=handles.t+handles.delta;
        handles.tetha=exp((handles.n2*handles.F*handles.alpha*(handles.Ei-
handles.E022)/(handles.R*handles.T)));
        handles.gama=sqrt(handles.Do2/handles.DR2);
        handles.tetgama=handles.tetha*handles.gama;

M=round((handles.n2*handles.F*handles.nu*handles.alpha*handles.t)/(handles.R*
handles.T*handles.delta));

handles.b=(handles.n2*handles.F*handles.alpha*handles.nu)/(handles.R*handles.
T);
        handles.k=handles.ks*exp(((-
handles.alpha*handles.n2*handles.F)/(handles.R*handles.T))*(handles.Ei-
handles.E022));
        handles.u=log(sqrt(pi*handles.Do2*handles.b)/handles.k);
        S4=zeros(M,M);
        for i=1:M;
            j=0;
            B4(i,1)=1/(2*sqrt(handles.delta)*(1+(handles.tetgama*exp(-
handles.delta*(i)))));
            while j<M;
                j=j+1;
```

```
                N=i;
                if j<N;
                    S4(i,j)=sqrt(N-(j-1))-sqrt(N-j);
                elseif N==j
                    S4(i,j)=1;
                else j>N;
                    S4(i,j)=0;
                end
            end
        end
        x4=S4\B4; % or x=inv(S)*B
        Xbt4=x4(i,1);

handles.I21irrev(1,w)=handles.n2*handles.F*handles.A*handles.C*sqrt(handles.D
R2*handles.b*pi)*Xbt4;

handles.E21irrev(1,w)=handles.E022+(((handles.R*handles.T)/(handles.F*handles
.n2*handles.alpha))*(handles.u-(handles.b*handles.t)))-
(((handles.R*handles.T)/(handles.F*handles.n2*handles.alpha))*log(sqrt(pi*han
dles.Do2*handles.b)/handles.ks));
    end
    handles.t=handles.trev2;
    for w=1:round(abs(handles.trev3-handles.t)/(2*handles.delta));
        handles.t=handles.t+handles.delta;
        handles.tetha=exp((handles.n2*handles.F*handles.alpha*(handles.Ei-
handles.E022)/(handles.R*handles.T)));
        handles.gama=sqrt(handles.Do2/handles.DR2);
        handles.tetgama=handles.tetha*handles.gama;

M=round((handles.n2*handles.F*handles.nu*handles.alpha*handles.t)/(handles.R*
handles.T*handles.delta));

handles.b=(handles.n2*handles.F*handles.alpha*handles.nu)/(handles.R*handles.
T);
        handles.k1=handles.ks*exp(((-
handles.alpha*handles.n2*handles.F)/(handles.R*handles.T))*(handles.Ei-
handles.E022));
        handles.u1=log(sqrt(pi*handles.Do2*handles.b)/handles.k1);
        S5=zeros(M,M);
        for i=1:M;
            j=0;

B5(i,1)=1/(2*sqrt(handles.delta)*(1+(handles.tetgama*exp(handles.delta*(i)-
(2*handles.landa*handles.delta)))));
            while j<M;
                j=j+1;
                N=i;
                if j<N;
                    S5(i,j)=sqrt(N-(j-1))-sqrt(N-j);
                elseif N==j
                    S5(i,j)=1;
                else j>N;
                    S5(i,j)=0;
                end
            end
        end
```

```
        x5=S5\B5; % or x=inv(S)*B
        Xbt5=x5(i,1);

handles.I2irrev(1,w)=handles.n2*handles.F*handles.A*handles.C*sqrt(handles.DR
2*handles.b*pi)*Xbt5;

handles.bt=((handles.alpha*handles.n2*handles.F)/(handles.R*handles.T))*((2*h
andles.nu*handles.landa)-(handles.nu*handles.t));

handles.E2irrev(1,w)=handles.E022+(((handles.R*handles.T)/(handles.F*handles.
n2*handles.alpha))*(handles.u1-(handles.bt)))-
(((handles.R*handles.T)/(handles.F*handles.n2*handles.alpha))*log(sqrt(pi*han
dles.Do2*handles.b)/handles.ks));
    end
    %% I vs E Plot and compare with Raw Data
    axis square
    ah1=subplot(1,2,1);
    plot(ah1,handles.E1rev,-handles.I1rev,'b.')
    hold on
    plot(ah1,handles.E12rev,handles.I12rev,'b.')
    hold on
    plot(ah1,handles.E1irrev,-handles.I1irrev,'r.')
    hold on
    plot(ah1,handles.E21irrev,handles.I21irrev,'r.')
    hold on
    plot(ah1,handles.E2irrev,handles.I2irrev,'r.')
    hold on
    xlabel('Potential Cathodic, E, v')
    ylabel('Current Cathodic, I, amp')
    hold on
    %% Concentration Plot Cathodic/Anodic side
    %Part 2: Irreversible Cathodic:
    handles.t=handles.trev1;
    handles.Co=handles.C;
    for w=1:round(abs(handles.trev2-handles.t)/(handles.delta));
        handles.t=handles.t+handles.delta;
        if w==1;
        handles.t2(1,1)=handles.trev1+handles.delta;
        elseif w>1;
        handles.t2(1,w)=handles.t2(1,w-1)+handles.delta;
        end
        handles.tetha=exp((handles.n2*handles.F*handles.alpha*(handles.Ei-
(handles.nu*handles.t)-handles.E02)/(handles.R*handles.T)));
        handles.gama=sqrt(handles.Do2/handles.DR2);
        handles.tetgama=handles.tetha*handles.gama;
        handles.Co1irrev(1,w)=handles.Co*(1-(1/(1+handles.tetgama)));
        handles.CR1irrev(1,w)=handles.Co-handles.Co1irrev(1,w);
    end
    % %Part 3: Irreversible Anodic:
    handles.t=handles.trev2;
    handles.CR21=handles.CR1irrev(1,w);
    for w=1:round(abs(handles.trev3-handles.t)/(handles.delta));
        handles.t=handles.t+handles.delta;
        if w==1;
        handles.t3(1,w)=handles.trev2+handles.delta;
        elseif w>1;
```

```
        handles.t3(1,w)=handles.t3(1,w-1)+handles.delta;
        end
        handles.tetha=exp((handles.n2*handles.F*handles.alpha*(handles.Ei-
(2*handles.nu*handles.landa)+(handles.nu*handles.t)-
handles.E022)/(handles.R*handles.T)));
        handles.gama=sqrt(handles.Do2/handles.DR2);
        handles.tetgama=handles.tetha*handles.gama;
        handles.CR21irrev(1,w)=handles.CR21*(1-
(handles.tetgama/(1+handles.tetgama)));
        handles.Co21irrev(1,w)=handles.CR21-handles.CR21irrev(1,w);
    end
    %Part 4: Reversible Anodic:
    handles.t=handles.trev3;
    handles.CR2=handles.Co21irrev(1,w);
    for w=1:round(abs(handles.trev3-handles.tfinal)/(handles.delta));
        handles.t=handles.t+handles.delta;
        if w==1;
        handles.t4(1,w)=handles.trev3+handles.delta;
        elseif w>1;
        handles.t4(1,w)=handles.t4(1,w-1)+handles.delta;
        end
        handles.tetha=exp((handles.n1*handles.F*(handles.Ei-
(2*handles.nu*handles.landa)+(handles.nu*handles.t)-
handles.E012)/(handles.R*handles.T)));
        handles.gama=sqrt(handles.Do1/handles.DR1);
        handles.tetgama=handles.tetha*handles.gama;
        handles.CR2rev(1,w)=handles.CR2*(1-
((handles.tetgama/(1+handles.tetgama))));
        handles.Co2rev(1,w)=handles.CR2-handles.CR2rev(1,w);
    end
    %Part 1: Reversible Cathodic:
    handles.t=handles.tinitial;
    handles.Co1=handles.Co2rev(1,w);
    for w=1:round(abs(handles.trev1-handles.t)/(handles.delta));
        handles.t=handles.t+handles.delta;
        if w==1;
        handles.t1(1,w)=handles.tinitial+handles.delta;
        elseif w>1;
        handles.t1(1,w)=handles.t1(1,w-1)+handles.delta;
        end
        handles.tetha=exp((handles.n1*handles.F*(handles.Ei-
(handles.nu*handles.t)-handles.E01)/(handles.R*handles.T)));
        handles.gama=sqrt(handles.Do1/handles.DR1);
        handles.tetgama=handles.tetha*handles.gama;
        handles.Co1rev(1,w)=handles.Co1*(1-(1/(1+handles.tetgama)));
        handles.CR1rev(1,w)=handles.Co1-handles.Co1rev(1,w);
    end
    axis square
    ah2=subplot(1,2,2);
    plot(ah2,handles.t1,handles.Co1rev*(handles.u+(4*handles.cl)),'b*')
    hold on
    plot(ah2,handles.t1,handles.CR1rev*(handles.u+(3*handles.cl)),'r*')
    hold on
    plot(ah2,handles.t2,handles.Co1irrev*(handles.u+(3*handles.cl)),'r*')
    hold on
    plot(ah2,handles.t2,handles.CR1irrev*handles.u,'K*')
```

```
    hold on
    plot(ah2,handles.t3,handles.Co21irrev*(handles.u+(3*handles.cl)),'r*')
    hold on
    plot(ah2,handles.t3,handles.CR21irrev*handles.u,'K*')
    hold on
    plot(ah2,handles.t4,handles.Co2rev*(handles.u+(4*handles.cl)),'b*')
    hold on
    plot(ah2,handles.t4,handles.CR2rev*(handles.u+(3*handles.cl)),'r*')
    hold on
    xlabel('Time (Second)')
    ylabel('Concentration (g/cm3)')

legend('U+4,Rev,C','U+3,Rev,C','U+3,Irrev,C','U,Irrev,C','U+3,Irrev,A','U,Irr
ev,A','U+4,Reve,A','U+3,Reve,A','Location','northeastoutside')

    % --- If Enable == 'on', executes on mouse press in 5 pixel border.
    % --- Otherwise, executes on mouse press in 5 pixel border or over text4.
    function text4_ButtonDownFcn(hObject, eventdata, handles)
    % hObject    handle to text4 (see GCBO)
    % eventdata reserved - to be defined in a future version of MATLAB
    % handles    structure with handles and user data (see GUIDATA)
```

Appendix V: Diffusion Model for Zirconium Chloride

The area ratio, current and potential errors are listed in Table V.1 to V.3. In addition, the diffusion coefficients, and formal potentials at B_c and C_c peaks for 1.07, 2.49, 4.98 wt% uranium chloride are listed below.

Table V.1 Area ratio, current and potential errors for 1.07 wt% uranium chloride at 250 mV/s.

Zr^{+4}	Zr^{+2}	**1.07 wt%- 250 mV/s**		
		Area Ratio	Current Error (%)	Potential Error (%)
100	0	0.7191	0.0851	0.0189
90	10	0.6995	0.1042	3.9124
80	20	0.6789	0.0268	3.9124
70	30	0.6597	0.1124	0.0189
60	40	0.6389	0.0502	0.0189
50	50	0.6597	0.1093	0.0189
40	60	0.6796	0.0598	0.0189
30	70	0.7006	0.0651	3.9124
20	80	0.7202	0.0669	0.0189
10	90	0.7387	0.0267	0.0189
0	100	0.7562	0.0860	0.0189

Table V.2 Area ratio, current and potential errors for 2.49 wt% uranium chloride at different scan rates.

Zr^{+4}	Zr^{+2}	2.49 wt%- 150 mV/s			2.49 wt%- 200 mV/s			2.49 wt%- 250 mV/s		
		Area Ratio	Current Error (%)	Potential Error (%)	Area Ratio	Current Error (%)	Potential Error (%)	Area Ratio	Current Error (%)	Potential Error (%)
100	0	1.107	0.102	1.825	1.125	0.297	1.301	1.078	0.065	5.625
90	10	1.085	0.050	1.825	1.108	0.412	6.243	1.055	0.149	4.447
80	20	1.062	0.011	2.671	1.078	0.027	6.243	1.035	0.487	4.447
70	**30**	**1.038**	**0.022**	**2.671**	**1.054**	**0.001**	**6.243**	**1.006**	**0.116**	**5.625**
60	40	1.016	0.040	2.671	1.028	0.025	6.243	0.985	0.030	4.447
50	50	0.989	0.060	2.671	1.000	0.091	6.243	0.958	0.112	4.447
40	60	0.965	0.009	1.825	0.973	0.137	1.301	0.933	0.101	4.447
30	70	0.938	0.156	1.820	0.949	0.016	6.243	0.909	0.067	4.447
20	80	0.913	0.128	1.825	0.921	0.033	1.301	0.882	0.123	5.625
10	90	0.886	0.099	2.671	0.892	0.105	1.301	0.856	0.098	5.625
0	100	0.856	0.093	2.671	0.863	0.064	6.243	0.828	0.028	4.447

Table V.3 Area ratio, current and potential errors for 4.98 wt% uranium chloride at different scan rates.

Zr^{+4}	Zr^{+2}	4.98 wt%- 100 mV/s			4.98 wt%- 200 mV/s			4.98 wt%- 300 mV/s		
		Area Ratio	Current Error (%)	Potential Error (%)	Area Ratio	Current Error (%)	Potential Error (%)	Area Ratio	Current Error (%)	Potential Error (%)
100	0	1.173	0.450	2.145	1.065	0.538	3.698	1.152	0.252	3.549
90	10	1.140	0.113	2.145	1.047	0.350	2.635	1.132	0.274	9.253
80	20	1.115	0.153	0.015	1.018	0.015	3.698	1.109	0.223	9.253
70	30	1.086	0.120	2.145	0.997	0.406	2.635	1.091	0.654	9.253
60	40	1.055	0.252	2.145	0.968	0.071	3.698	1.052	0.388	9.253
50	50	1.027	0.248	2.145	0.941	0.260	2.635	1.030	0.182	9.253
40	60	0.998	0.313	2.145	0.915	0.173	2.635	1.009	0.221	9.253
30	**70**	**0.972**	**0.022**	**2.145**	**0.892**	**0.086**	**3.69**	**0.982**	**0.033**	**3.549**
20	80	0.941	0.074	2.145	0.862	0.145	3.698	0.951	0.074	3.549
10	90	0.911	0.092	2.145	0.836	0.056	3.698	0.923	0.116	9.253
0	100	0.881	0.090	2.145	0.808	0.099	3.698	0.892	0.153	3.549

Table V.4 Diffusion coefficients and formal potentials for $\mathbf{B_c}$ and $\mathbf{C_c}$ peaks for 1.07, 2.48, and 4.98 wt% $ZrCl_4$ at different scan rates at 773 K.

Diffusion Coefficient × 10^5 (cm^2/s)												
Peak, Rxn	1.07 wt%			2.49 wt%					4.98 wt%			
	250 mV/s	300 mV/s	350 mV/s	100 mV/s	150 mV/s	200 mV/s	250 mV/s	300 mV/s	100 mV/s	150 mV/s	200 mV/s	300 mV/s
B_c Zr^{+2}/Zr	4.65	4.41	4.09	4.27	4.05	3.87	3.56	3.32	2.107	2.15	2.07	2.01
C_c Zr^{+4}/Zr	0.78	0.82	0.72	1.64	1.02	0.88	0.68	0.72	0.52	0.26	0.26	0.25
Formal Potential (V)												
B_c Zr^{+2}/Zr	-1.44	-1.49	-1.50	-1.51	-1.54	-1.55	-1.62	-1.64	-1.68	-1.76	-1.85	-1.94
C_c Zr^{+4}/Zr	-1.85	-1.84	-1.88	-2.04	-2.04	2.061	-2.09	-2.14	-2.17	-1.95	-2.01	-2.06

Appendix VI: ANI *Matlab* and GUI Codes

VI.1 *Matlab* Code

VI.1.1 Zirconium Chloride

```
%Train ANI to predict current (A)
%Dimensions of the CSV =
%Potential (V), Time (s), Weight (%), ScanRate (data_inputmv/s), Current (A)
clc
clear all;
close all;

%predefined variables

data = load('Zr_data_4_v4.csv');
num_training_data = 33660;

[data_row data_col] = size (data);
num_data_points = data_row;
num_dim = data_col;

%Break the data in to input and output
%The last dimension is the output
data_input = data(:, 1:num_dim - 1);
data_output = data(:, num_dim);

%Need to use the transpose
data_input = data_input';
data_output = data_output';

%Scale the data between 0 and 1
[data_input_norm, in_ps] = mapminmax(data_input);
[data_output_norm, out_ps] = mapminmax(data_output);

%Separate the traning and testing data
tr_data_input  = data_input_norm(:,1:num_training_data);
tr_data_output = data_output_norm(:,1:num_training_data);

te_data_input  = data_input_norm(:,num_training_data+1:num_data_points);
te_data_output = data_output_norm(:,num_training_data+1:num_data_points);

jj=0;
LL={0};
S=[1 0]; %S=[1] for one hidden layer, S=[1 0] for two hidden layers
Num_layer=0;
```

```
Num_Neuron={0};
Num_Validation=0;
%Artificial Neural Intelligence
for k=1:30
    S(end)=k;
net = newff(tr_data_input,tr_data_output,S);
%ANI parameters
for j=1:30

    jj=jj+1;
    Num_Neuron{jj,1}=num2str(S);
    Num_layer(jj,1)=ii;
    Num_Validation(jj,1)=j;
net.trainParam.min_grad = 0;
net.trainParam.epochs=5000;
net.trainParam.max_fail =j;
% Training
[trained_net,tr] = train(net,tr_data_input,tr_data_output);

%Testing
[test_net_out,Pf,Af,E,perf] = sim(trained_net,te_data_input);

%Rescale the outputs
test_outputs_original = mapminmax('reverse',te_data_output,out_ps);
test_net_out_original = mapminmax('reverse',test_net_out,out_ps);

test_outputs_original = test_outputs_original';
test_net_out_original = test_net_out_original';
%%Simulating over all sequenced data

%Break the data in to input and output
%The last dimension is the output
data_seq_input = data(:, 1:num_dim - 1)';
data_seq_output = data(:, num_dim)';

%Scale the data between 0 and 1
data_seq_input_norm  = mapminmax('apply', data_seq_input, in_ps);
data_seq_output_norm = mapminmax('apply', data_seq_output, out_ps);

%Running the ANI
[net_seq_out,Pf,Af,E,perf] = sim(trained_net,data_seq_input_norm);

%Rescale the outputs
net_seq_out_original = mapminmax('reverse',net_seq_out,out_ps);

%Combining all into one matrix
temp_write_mat = [data_seq_input' data_seq_output' net_seq_out_original'];
csvwrite('predicted_values_repeat_new.csv', temp_write_mat);
%0.5 wt% concentration with 200 mV/s
data_Pointfive_train= temp_write_mat(temp_write_mat(:, 3) == 0.5, :);
data_Point_five_train=data_Pointfive_train(data_Pointfive_train(:,4)==200,:);
[data_row_temp_train data_col_temp_train] = size (data_Point_five_train);
for i=1:data_row_temp_train
error_Point_five_train(i,1) = abs((data_Point_five_train(i,5) -
data_Point_five_train(i,6))./data_Point_five_train(i,6));
```

```
end
%0.5 wt% concentration with 450 mV/s
data_Pointfive_test= temp_write_mat(temp_write_mat(:, 3) == 0.5, :);
data_Point_five_test=data_Pointfive_test(data_Pointfive_test(:,4)==450,:);
[data_row_temp_test data_col_temp_test] = size (data_Point_five_test);
for i=1:data_row_temp_test
error_Point_five_test(i,1) = abs((data_Point_five_test(i,5) -
data_Point_five_test(i,6))./data_Point_five_test(i,6));
end
Ave_abs_Point_five_train(jj,1) = mean(error_Point_five_train(:,1))*100;
Ave_abs_Point_five_test(jj,1) = mean(error_Point_five_test(:,1))*100;
rowname=[1:jj];
R=strsplit(num2str(rowname),' ');
end
end
Result=[Num_Validation Ave_abs_Point_five_train Ave_abs_Point_five_test];
Table=table(Num_layer,Num_Neuron,Num_Validation,Ave_abs_Point_five_train,Ave_
abs_Point_five_test,...
    'RowNames',R)
```

VI.1.2 Uranium Chloride

```
%Train ANI to predict current (A)
%Dimensions of the CSV =
%Potential (V), Time (s), Weight (%), ScanRate (mv/s), Current (A)
clc
clear all;
close all;

%Predefined variables

data = load('u-dataset-ANI-V5.csv');
num_training_data = 201287;

[data_row data_col] = size (data);
num_data_points = data_row;
num_dim = data_col;

%Break the data in to input and output
%The last dimension is the output
data_input = data(:, 1:num_dim - 1);
data_output = data(:, num_dim);

%Need to use the transpose
data_input = data_input';
data_output = data_output';

%Scale the data between 0 and 1
[data_input_norm, in_ps] = mapminmax(data_input);
[data_output_norm, out_ps] = mapminmax(data_output);

%Separate the traning and testing data
```

```
tr_data_input  = data_input_norm(:,1:num_training_data);
tr_data_output = data_output_norm(:,1:num_training_data);

te_data_input  = data_input_norm(:,num_training_data+1:num_data_points);
te_data_output = data_output_norm(:,num_training_data+1:num_data_points);

jj=0;
LL={0};
ii=1; %Number of Layer
S=[1]; % S=[1] One Hidden Layer, S=[1 0] two hidden layer
dd=0;
Num_layer=0;
Num_Neuron={0};
Num_Validation=0;
%Artificial Neural Intelligence
for k=1:30
    S(end)=k;
net = newff(tr_data_input,tr_data_output,S);
%ANI parameters
for j=1:30
    dd=dd+1;
    jj=jj+1;
    Num_Neuron{jj,1}=num2str(S);
    Neuron(jj,1)=k;
    Num_layer(jj,1)=ii;
    Num_Validation(jj,1)=j;
net.trainParam.min_grad = 0;
net.trainParam.epochs=5000;
net.trainParam.max_fail =j;
%Training
[trained_net,tr] = train(net,tr_data_input,tr_data_output);

%Testing
[test_net_out,Pf,Af,E,perf] = sim(trained_net,te_data_input);

%Rescale the outputs
test_outputs_original = mapminmax('reverse',te_data_output,out_ps);
test_net_out_original = mapminmax('reverse',test_net_out,out_ps);

test_outputs_original = test_outputs_original';
test_net_out_original = test_net_out_original';
%%Simulating over all sequenced data

%Break the data in to input and output
%the last dimension is the output
data_seq_input = data(:, 1:num_dim - 1)';
data_seq_output = data(:, num_dim)';

%Scale the data between 0 and 1
data_seq_input_norm  = mapminmax('apply', data_seq_input, in_ps);
data_seq_output_norm = mapminmax('apply', data_seq_output, out_ps);

%Running the ANI
[net_seq_out,Pf,Af,E,perf] = sim(trained_net,data_seq_input_norm);
```

```
%Rescale the outputs
net_seq_out_original = mapminmax('reverse',net_seq_out,out_ps);

%Combining all into one matrix
temp_write_mat = [data_seq_input' data_seq_output' net_seq_out_original'];
filename=sprintf('Predicted_Values_Second_Layer_Uranium_%d.csv',dd);
csvwrite(filename, temp_write_mat);
%% 0.5 wt% concentration

%Scan Rate= 150 mV/s
data_pointfive_test_1= temp_write_mat(temp_write_mat(:, 3) == 0.5, :);
data_pointfive_test_1=data_pointfive_test_1(data_pointfive_test_1(:,4)==150,:
);
[data_row_temp_test data_col_temp_test] = size (data_pointfive_test_1);
for i=1:data_row_temp_test
error_pointfive_test_1(i,1) = abs((data_pointfive_test_1(i,5) -
data_pointfive_test_1(i,6))./data_pointfive_test_1(i,6));
end
error_pointfive_150(jj,1)=mean(error_pointfive_test_1(:,1))*100;

%Scan Rate= 300 mV/s
data_pointfive_test_2= temp_write_mat(temp_write_mat(:, 3) == 0.5, :);
data_pointfive_test_2=data_pointfive_test_2(data_pointfive_test_2(:,4)==300,:
);
[data_row_temp_test data_col_temp_test] = size (data_pointfive_test_2);
for i=1:data_row_temp_test
error_pointfive_test_2(i,1) = abs((data_pointfive_test_2(i,5) -
data_pointfive_test_2(i,6))./data_pointfive_test_2(i,6));
end
error_pointfive_300(jj,1)=mean(error_pointfive_test_2(:,1))*100;

rowname=[1:jj];
R=strsplit(num2str(rowname),' ');
end
Result=[Num_Validation error_pointfive_150 error_pointfive_300];
end
Table=table(Num_layer,Num_Neuron,Num_Validation,error_pointfive_150,error_poi
ntfive_300,...
    'RowNames',R)
```

VI.2 GUI Code

VI.2.1 Zirconium Chloride

```
function varargout = ANI_GUI(varargin)
% ANI_GUI MATLAB code for ANI_GUI.fig
%      ANI_GUI, by itself, creates a new ANI_GUI or raises the existing
%      singleton*.
%
%      H = ANI_GUI returns the handle to a new ANI_GUI or the handle to
```

```
%      the existing singleton*.
%
%      ANI_GUI('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in ANI_GUI.M with the given input arguments.
%
%      ANI_GUI('Property','Value',...) creates a new ANI_GUI or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before ANI_GUI_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to ANI_GUI_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help ANI_GUI

% Last Modified by GUIDE v2.5 10-May-2017 12:51:10

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @ANI_GUI_OpeningFcn, ...
                   'gui_OutputFcn',  @ANI_GUI_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before ANI_GUI is made visible.
function ANI_GUI_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to ANI_GUI (see VARARGIN)

% Choose default command line output for ANI_GUI
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes ANI_GUI wait for user response (see UIRESUME)
```

```
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = ANI_GUI_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;



function edit5_Callback(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str1 = get(hObject, 'String');
val1 = str2num(str1);
handles.S1= val1;
guidata(hObject,handles);

% Hints: get(hObject,'String') returns contents of edit5 as text
%        str2double(get(hObject,'String')) returns contents of edit5 as a
double


% --- Executes during object creation, after setting all properties.
function edit5_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit6_Callback(hObject, eventdata, handles)
% hObject    handle to edit6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str2 = get(hObject, 'String');
val2 = str2num(str2);
handles.S2= val2;
guidata(hObject,handles);

% Hints: get(hObject,'String') returns contents of edit6 as text
```

```
%        str2double(get(hObject,'String')) returns contents of edit6 as a
double


% --- Executes during object creation, after setting all properties.
function edit6_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit7_Callback(hObject, eventdata, handles)
% hObject    handle to edit7 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str3 = get(hObject, 'String');
val3 = str2num(str3);
handles.S3= val3;
guidata(hObject,handles);

% Hints: get(hObject,'String') returns contents of edit7 as text
%        str2double(get(hObject,'String')) returns contents of edit7 as a
double


% --- Executes during object creation, after setting all properties.
function edit7_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit7 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit8_Callback(hObject, eventdata, handles)
% hObject    handle to edit8 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str5 = get(hObject, 'String');
val5 = str2num(str5);
```

```
handles.j= val5; % Validation Number
guidata(hObject,handles);

% Hints: get(hObject,'String') returns contents of edit8 as text
%        str2double(get(hObject,'String')) returns contents of edit8 as a
double


% --- Executes during object creation, after setting all properties.
function edit8_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit8 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit9_Callback(hObject, eventdata, handles)
% hObject    handle to edit9 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str6 = get(hObject, 'String');
val6 = str2num(str6);
handles.W1= val6;
guidata(hObject,handles);

% Hints: get(hObject,'String') returns contents of edit9 as text
%        str2double(get(hObject,'String')) returns contents of edit9 as a
double


% --- Executes during object creation, after setting all properties.
function edit9_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit9 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
```

```
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
handles.data = load('Zr_data_4_v4.csv');
handles.num_training_data = 33660;

[handles.data_row handles.data_col] = size (handles.data);
handles.num_data_points = handles.data_row;
handles.num_dim = handles.data_col;

%Break the data in to input and output
%the last dimension is the output
handles.data_input = handles.data(:, 1:handles.num_dim - 1);
handles.data_output = handles.data(:, handles.num_dim);

% need to use the transpose
handles.data_input = handles.data_input';
handles.data_output = handles.data_output';

%scale the data between 0 and 1
[handles.data_input_norm, handles.in_ps] = mapminmax(handles.data_input);
[handles.data_output_norm, handles.out_ps] = mapminmax(handles.data_output);

%separate the traning and testing data
handles.tr_data_input  =
handles.data_input_norm(:,1:handles.num_training_data);
handles.tr_data_output =
handles.data_output_norm(:,1:handles.num_training_data);

handles.te_data_input  =
handles.data_input_norm(:,handles.num_training_data+1:handles.num_data_points
);
handles.te_data_output =
handles.data_output_norm(:,handles.num_training_data+1:handles.num_data_point
s);

handles.jj=0;
handles.LL={0};
handles.S=[handles.S1 handles.S2 handles.S3];
if handles.S2==0;
handles.ii=1; %Number of Layers
handles.S=[handles.S1];
elseif handles.S3==0;
    handles.ii=2;
    handles.S=[handles.S1 handles.S2];
elseif handles.S4==0;
    handles.ii=3;
    handles.S=[handles.S1 handles.S2 handles.S3];
end
handles.jj=0;
handles.LL=0;
handles.layer=0;
handles.Neuron={0};
handles.Validation=0;
% Artificial Neural Network
handles.net = newff(handles.tr_data_input,handles.tr_data_output,handles.S);
```

```
% ANI parameters
handles.jj=handles.jj+1;
handles.Neuron{handles.jj,1}=num2str(handles.S);
handles.layer(handles.jj,1)=handles.ii;
handles.Validation(handles.jj,1)=handles.j;
handles.net.trainParam.min_grad = 0;
handles.net.trainParam.epochs=5000;
handles.net.trainParam.max_fail =handles.j;
% Training
[handles.trained_net,handles.tr] =
train(handles.net,handles.tr_data_input,handles.tr_data_output);
handles.epoch=handles.tr.epoch;
handles.t=handles.tr.time;
% testing
[handles.test_net_out,handles.Pf,handles.Af,handles.E,handles.perf] =
sim(handles.trained_net,handles.te_data_input);
% rescale the outputs
handles.test_outputs_original =
mapminmax('reverse',handles.te_data_output,handles.out_ps);
handles.test_net_out_original =
mapminmax('reverse',handles.test_net_out,handles.out_ps);

handles.test_outputs_original = handles.test_outputs_original';
handles.test_net_out_original = handles.test_net_out_original';
% Simulating over all sequenced data
%Break the data in to input and output
%the last dimension is the output
handles.data_seq_input = handles.data(:, 1:handles.num_dim - 1)';
handles.data_seq_output = handles.data(:, handles.num_dim)';

%scale the data between 0 and 1
handles.data_seq_input_norm  = mapminmax('apply', handles.data_seq_input,
handles.in_ps);
handles.data_seq_output_norm = mapminmax('apply', handles.data_seq_output,
handles.out_ps);

% running the ANI
[handles.net_seq_out,handles.Pf,handles.Af,handles.E,handles.perf] =
sim(handles.trained_net,handles.data_seq_input_norm);

% rescale the outputs
handles.net_seq_out_original =
mapminmax('reverse',handles.net_seq_out,handles.out_ps);

%combining all into one matrix
handles.temp_write_mat = [handles.data_seq_input' handles.data_seq_output'
handles.net_seq_out_original'];
csvwrite('predicted_values_GUI.csv', handles.temp_write_mat);
%% 0.5 wt% concentration with 200 mV/s:
handles.data_Pointfive_train=
handles.temp_write_mat(handles.temp_write_mat(:, 3) == 0.5, :);
handles.data_Point_five_train=handles.data_Pointfive_train(handles.data_Point
five_train(:,4)==200,:);
[handles.data_row_temp_train handles.data_col_temp_train] = size
(handles.data_Point_five_train);
```

```
for i=1:handles.data_row_temp_train
handles.error_Point_five_train(i,1) = abs((handles.data_Point_five_train(i,5)
- handles.data_Point_five_train(i,6))./handles.data_Point_five_train(i,6));
end
handles.error_point_five_200=mean(handles.error_Point_five_train(:,1))*100;
%% 0.5 wt% concentration with 450 mV/s:
handles.data_Pointfive_test= handles.temp_write_mat(handles.temp_write_mat(:,
3) == 0.5, :);
handles.data_Point_five_test=handles.data_Pointfive_test(handles.data_Pointfi
ve_test(:,4)==450,:);
[handles.data_row_temp_test handles.data_col_temp_test] = size
(handles.data_Point_five_test);
for i=1:handles.data_row_temp_test
handles.error_Point_five_test(i,1) = abs((handles.data_Point_five_test(i,5) -
handles.data_Point_five_test(i,6))./handles.data_Point_five_test(i,6));
end
handles.error_point_five_450=mean(handles.error_Point_five_test(:,1))*100;
%% 1 wt% concentration with 200 mV/s:
handles.data_Pointone_test_1=
handles.temp_write_mat(handles.temp_write_mat(:, 3) == 1, :);
handles.data_Point_one_test_1=handles.data_Pointone_test_1(handles.data_Point
one_test_1(:,4)==200,:);
[handles.data_row_one_test_1 handles.data_col_one_test_1] = size
(handles.data_Point_one_test_1);
for i=1:handles.data_row_one_test_1
handles.error_one_test_200(i,1) = abs((handles.data_Point_one_test_1(i,5) -
handles.data_Point_one_test_1(i,6))./handles.data_Point_one_test_1(i,6));
end
handles.error_one_200=mean(handles.error_one_test_200(:,1))*100;
%% 1 wt% concentration with 350 mV/s:
handles.data_Pointone_test_2=
handles.temp_write_mat(handles.temp_write_mat(:, 3) == 1, :);
handles.data_Point_one_test_2=handles.data_Pointone_test_2(handles.data_Point
one_test_2(:,4)==350,:);
[handles.data_row_one_test_2 handles.data_col_one_test_2] = size
(handles.data_Point_one_test_2);
for i=1:handles.data_row_one_test_2
handles.error_one_test_350(i,1) = abs((handles.data_Point_one_test_2(i,5) -
handles.data_Point_one_test_2(i,6))./handles.data_Point_one_test_2(i,6));
end
handles.error_one_350=mean(handles.error_one_test_350(:,1))*100;
%% 2.5 wt% concentration with 150 mV/s:
handles.data_Pointtwo_test_1=
handles.temp_write_mat(handles.temp_write_mat(:, 3) == 2.5, :);
handles.data_Point_two_test_1=handles.data_Pointtwo_test_1(handles.data_Point
two_test_1(:,4)==150,:);
[handles.data_row_two_test_1 handles.data_col_two_test_1] = size
(handles.data_Point_two_test_1);
for i=1:handles.data_row_two_test_1
handles.error_two_test_150(i,1) = abs((handles.data_Point_two_test_1(i,5) -
handles.data_Point_two_test_1(i,6))./handles.data_Point_two_test_1(i,6));
end
handles.error_twopintfive_150=mean(handles.error_two_test_150(:,1))*100;
%% 2.5 wt% concentration with 200 mV/s:
handles.data_Pointtwo_test_2=
handles.temp_write_mat(handles.temp_write_mat(:, 3) == 2.5, :);
```

```
handles.data_Point_two_test_2=handles.data_Pointtwo_test_2(handles.data_Point
two_test_2(:,4)==200,:);
[handles.data_row_two_test_2 handles.data_col_two_test_2] = size
(handles.data_Point_two_test_2);
for i=1:handles.data_row_two_test_2
handles.error_two_test_200(i,1) = abs((handles.data_Point_two_test_2(i,5) -
handles.data_Point_two_test_2(i,6))./handles.data_Point_two_test_2(i,6));
end
handles.error_twopointfive_200=mean(handles.error_two_test_200(:,1))*100;
%% 2.5 wt% concentration with 400 mV/s:
handles.data_Pointtwo_test_3=
handles.temp_write_mat(handles.temp_write_mat(:, 3) == 2.5, :);
handles.data_Point_two_test_3=handles.data_Pointtwo_test_3(handles.data_Point
two_test_3(:,4)==400,:);
[handles.data_row_two_test_3 handles.data_col_two_test_3] = size
(handles.data_Point_two_test_3);
for i=1:handles.data_row_two_test_3
handles.error_two_test_400(i,1) = abs((handles.data_Point_two_test_3(i,5) -
handles.data_Point_two_test_3(i,6))./handles.data_Point_two_test_3(i,6));
end
handles.error_twopointfive_400=mean(handles.error_two_test_400(:,1))*100;
%% 5 wt% concentration with 100 mV/s:
handles.data_Pointfive_test_1=
handles.temp_write_mat(handles.temp_write_mat(:, 3) == 5, :);
handles.data_Point_five_test_1=handles.data_Pointfive_test_1(handles.data_Poi
ntfive_test_1(:,4)==100,:);
[handles.data_row_five_test_1 handles.data_col_five_test_1] = size
(handles.data_Point_five_test_1);
for i=1:handles.data_row_five_test_1
handles.error_five_test_100(i,1) = abs((handles.data_Point_five_test_1(i,5) -
handles.data_Point_five_test_1(i,6))./handles.data_Point_five_test_1(i,6));
end
handles.error_five_100=mean(handles.error_five_test_100(:,1))*100;
%% 5 wt% concentration with 250 mV/s:
handles.data_Pointfive_test_2=
handles.temp_write_mat(handles.temp_write_mat(:, 3) == 5, :);
handles.data_Point_five_test_2=handles.data_Pointfive_test_2(handles.data_Poi
ntfive_test_2(:,4)==250,:);
[handles.data_row_five_test_2 handles.data_col_five_test_2] = size
(handles.data_Point_five_test_2);
for i=1:handles.data_row_five_test_2
handles.error_five_test_250(i,1) = abs((handles.data_Point_five_test_2(i,5) -
handles.data_Point_five_test_2(i,6))./handles.data_Point_five_test_2(i,6));
end
handles.error_five_250=mean(handles.error_five_test_250(:,1))*100;
%% Table:
f = figure('Position', [100 100 752 250]);
handles.H={handles.error_point_five_200,handles.error_point_five_450,handles.
error_one_200,handles.error_one_350,handles.error_twopintfive_150,handles.err
or_twopointfive_200,handles.error_twopointfive_400,handles.error_five_100,han
dles.error_five_250};
t = uitable('Parent', f, 'Position', [25 50 700 200], 'Data', handles.H);
t.ColumnName={'0.5 wt% at 200 mV/s','0.5 wt% @ 450 mV/s','1 wt% @ 200
mV/s','1 wt% @ 350 mV/s','2.5 wt% @ 150 mV/s','2.5 wt% @ 200 mV/s','2.5 wt% @
400 mV/s','5 wt% @ 100 mV/s','2.5 wt% @ 250 mV/s'};
t.RowName = {'Error %'};
```

```
%% Figures
handles.data2 = load('predicted_values_GUI.csv');

handles.weight = unique(handles.data2(:,3));
handles.scan_rate = unique(handles.data2(:,4));

handles.num_weight = length(handles.weight);
handles.num_sr = length(handles.scan_rate);

handles.data_temp = handles.data2(handles.data2(:, 3) == handles.W1, :)
    for j=1:handles.num_sr
handles.data_temp_2 = handles.data_temp(handles.data_temp(:, 4) ==
handles.scan_rate(j), :);
        if(~isempty(handles.data_temp_2))
            figure;
            hold on;
            title(strcat(' (w = ', num2str(handles.W1), '%, Scanrate = ',
num2str(handles.scan_rate(j)), ' mV/s)'), 'FontWeight','bold');
            plot(handles.data_temp_2(:,1), handles.data_temp_2(:,5), '-b',
'Linewidth', 1.5);
            plot(handles.data_temp_2(:,1), handles.data_temp_2(:,6), '-r',
'Linewidth', 1.5);
            legend('Experimental (Hoover)', 'ANI','Location','northwest');
            xlabel('Potential (V)');
            ylabel('Current (amp)');
            grid on;
            hold off;
        end
    end
```

VI.2.1 Uranium Chloride

```
function varargout = ANI_GUI_U(varargin)
% ANI_GUI_U MATLAB code for ANI_GUI_U.fig
%      ANI_GUI_U, by itself, creates a new ANI_GUI_U or raises the existing
%      singleton*.
%
%      H = ANI_GUI_U returns the handle to a new ANI_GUI_U or the handle to
%      the existing singleton*.
%
%      ANI_GUI_U('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in ANI_GUI_U.M with the given input arguments.
%
%      ANI_GUI_U('Property','Value',...) creates a new ANI_GUI_U or raises
the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before ANI_GUI_U_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to ANI_GUI_U_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES
```

```
% Edit the above text to modify the response to help ANI_GUI_U

% Last Modified by GUIDE v2.5 10-May-2017 13:36:15

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @ANI_GUI_U_OpeningFcn, ...
                   'gui_OutputFcn',  @ANI_GUI_U_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before ANI_GUI_U is made visible.
function ANI_GUI_U_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to ANI_GUI_U (see VARARGIN)

% Choose default command line output for ANI_GUI_U
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes ANI_GUI_U wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = ANI_GUI_U_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;
```

```
function edit5_Callback(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str1 = get(hObject, 'String');
val1 = str2num(str1);
handles.S1= val1;
guidata(hObject,handles);

% Hints: get(hObject,'String') returns contents of edit5 as text
%        str2double(get(hObject,'String')) returns contents of edit5 as a
double


% --- Executes during object creation, after setting all properties.
function edit5_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit6_Callback(hObject, eventdata, handles)
% hObject    handle to edit6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str2 = get(hObject, 'String');
val2 = str2num(str2);
handles.S2= val2;
guidata(hObject,handles);

% Hints: get(hObject,'String') returns contents of edit6 as text
%        str2double(get(hObject,'String')) returns contents of edit6 as a
double


% --- Executes during object creation, after setting all properties.
function edit6_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
```

```
    set(hObject,'BackgroundColor','white');
end

function edit7_Callback(hObject, eventdata, handles)
% hObject    handle to edit7 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str3 = get(hObject, 'String');
val3 = str2num(str3);
handles.S3= val3;
guidata(hObject,handles);

% Hints: get(hObject,'String') returns contents of edit7 as text
%        str2double(get(hObject,'String')) returns contents of edit7 as a
double


% --- Executes during object creation, after setting all properties.
function edit7_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit7 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit8_Callback(hObject, eventdata, handles)
% hObject    handle to edit8 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str5 = get(hObject, 'String');
val5 = str2num(str5);
handles.j= val5; % Validation Number
guidata(hObject,handles);

% Hints: get(hObject,'String') returns contents of edit8 as text
%        str2double(get(hObject,'String')) returns contents of edit8 as a
double


% --- Executes during object creation, after setting all properties.
function edit8_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit8 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit9_Callback(hObject, eventdata, handles)
% hObject    handle to edit9 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str6 = get(hObject, 'String');
val6 = str2num(str6);
handles.W1= val6;
guidata(hObject,handles);

% Hints: get(hObject,'String') returns contents of edit9 as text
%        str2double(get(hObject,'String')) returns contents of edit9 as a
double


% --- Executes during object creation, after setting all properties.
function edit9_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit9 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
handles.data = load('u-dataset-ANI-V5.csv');
handles.num_training_data = 201287;


[handles.data_row handles.data_col] = size (handles.data);
handles.num_data_points = handles.data_row;
handles.num_dim = handles.data_col;

%Break the data in to input and output
%the last dimension is the output
handles.data_input = handles.data(:, 1:handles.num_dim - 1);
handles.data_output = handles.data(:, handles.num_dim);

% need to use the transpose
```

```
handles.data_input = handles.data_input';
handles.data_output = handles.data_output';

%scale the data between 0 and 1
[handles.data_input_norm, handles.in_ps] = mapminmax(handles.data_input);
[handles.data_output_norm, handles.out_ps] = mapminmax(handles.data_output);

%separate the traning and testing data
handles.tr_data_input  =
handles.data_input_norm(:,1:handles.num_training_data);
handles.tr_data_output =
handles.data_output_norm(:,1:handles.num_training_data);

handles.te_data_input  =
handles.data_input_norm(:,handles.num_training_data+1:handles.num_data_points
);
handles.te_data_output =
handles.data_output_norm(:,handles.num_training_data+1:handles.num_data_point
s);

handles.jj=0;
handles.LL={0};
handles.S=[handles.S1 handles.S2 handles.S3];
if handles.S2==0;
handles.ii=1; %Number of Layers
handles.S=[handles.S1];
elseif handles.S3==0;
    handles.ii=2;
    handles.S=[handles.S1 handles.S2];
elseif handles.S4==0;
    handles.ii=3;
    handles.S=[handles.S1 handles.S2 handles.S3];
end
handles.jj=0;
handles.LL=0;
handles.layer=0;
handles.Neuron={0};
handles.Validation=0;
% Artificial Neural Network
handles.net = newff(handles.tr_data_input,handles.tr_data_output,handles.S);
% ANI parameters
handles.jj=handles.jj+1;
handles.Neuron{handles.jj,1}=num2str(handles.S);
handles.layer(handles.jj,1)=handles.ii;
handles.Validation(handles.jj,1)=handles.j;
handles.net.trainParam.min_grad = 0;
handles.net.trainParam.epochs=5000;
handles.net.trainParam.max_fail =handles.j;
% Training
[handles.trained_net,handles.tr] =
train(handles.net,handles.tr_data_input,handles.tr_data_output);
handles.epoch=handles.tr.epoch;
handles.t=handles.tr.time;
% testing
[handles.test_net_out,handles.Pf,handles.Af,handles.E,handles.perf] =
sim(handles.trained_net,handles.te_data_input);
```

```
% rescale the outputs
handles.test_outputs_original =
mapminmax('reverse',handles.te_data_output,handles.out_ps);
handles.test_net_out_original =
mapminmax('reverse',handles.test_net_out,handles.out_ps);

handles.test_outputs_original = handles.test_outputs_original';
handles.test_net_out_original = handles.test_net_out_original';
% Simulating over all sequenced data
%Break the data in to input and output
%the last dimension is the output
handles.data_seq_input = handles.data(:, 1:handles.num_dim - 1)';
handles.data_seq_output = handles.data(:, handles.num_dim)';

%scale the data between 0 and 1
handles.data_seq_input_norm  = mapminmax('apply', handles.data_seq_input,
handles.in_ps);
handles.data_seq_output_norm = mapminmax('apply', handles.data_seq_output,
handles.out_ps);

% running the ANI
[handles.net_seq_out,handles.Pf,handles.Af,handles.E,handles.perf] =
sim(handles.trained_net,handles.data_seq_input_norm);

% rescale the outputs
handles.net_seq_out_original =
mapminmax('reverse',handles.net_seq_out,handles.out_ps);

%combining all into one matrix
handles.temp_write_mat = [handles.data_seq_input' handles.data_seq_output'
handles.net_seq_out_original'];
csvwrite('predicted_values_GUI.csv', handles.temp_write_mat);
%% 0.5 wt% concentration with 150 mV/s:
handles.data_Pointfive_train=
handles.temp_write_mat(handles.temp_write_mat(:, 3) == 0.5, :);
handles.data_Point_five_train=handles.data_Pointfive_train(handles.data_Point
five_train(:,4)==150,:);
[handles.data_row_temp_train handles.data_col_temp_train] = size
(handles.data_Point_five_train);
for i=1:handles.data_row_temp_train
handles.error_Point_five_train(i,1) = abs((handles.data_Point_five_train(i,5)
- handles.data_Point_five_train(i,6))./handles.data_Point_five_train(i,6));
end
handles.error_point_five_150=mean(handles.error_Point_five_train(:,1))*100;
%% 0.5 wt% concentration with 300 mV/s:
handles.data_Pointfive_test= handles.temp_write_mat(handles.temp_write_mat(:,
3) == 0.5, :);
handles.data_Point_five_test=handles.data_Pointfive_test(handles.data_Pointfi
ve_test(:,4)==300,:);
[handles.data_row_temp_test handles.data_col_temp_test] = size
(handles.data_Point_five_test);
for i=1:handles.data_row_temp_test
handles.error_Point_five_test(i,1) = abs((handles.data_Point_five_test(i,5) -
handles.data_Point_five_test(i,6))./handles.data_Point_five_test(i,6));
end
```

```
handles.error_point_five_300=mean(handles.error_Point_five_test(:,1))*100;

%% Table:
f = figure('Position', [100 100 752 250]);
handles.H={handles.error_point_five_150,handles.error_point_five_300};
t = uitable('Parent', f, 'Position', [25 50 700 200], 'Data', handles.H);
t.ColumnName={'0.5 wt% at 150 mV/s','0.5 wt% @ 300 mV/s'};
t.RowName = {'Error %'};
%% Figures
handles.data2 = load('predicted_values_GUI.csv');

handles.weight = unique(handles.data2(:,3));
handles.scan_rate = unique(handles.data2(:,4));

handles.num_weight = length(handles.weight);
handles.num_sr = length(handles.scan_rate);

handles.data_temp = handles.data2(handles.data2(:, 3) == handles.W1, :)
    for j=1:handles.num_sr
handles.data_temp_2 = handles.data_temp(handles.data_temp(:, 4) ==
handles.scan_rate(j), :);
        if(~isempty(handles.data_temp_2))
            figure;
            hold on;
            title(strcat(' (w = ', num2str(handles.W1), '%, Scanrate = ',
num2str(handles.scan_rate(j)), ' mV/s)'), 'FontWeight','bold');
            plot(handles.data_temp_2(:,1), handles.data_temp_2(:,5), '-b',
'Linewidth', 1.5);
            plot(handles.data_temp_2(:,1), handles.data_temp_2(:,6), '-r',
'Linewidth', 1.5);
            legend('Experimental (Hoover)', 'ANI','Location','northwest');
            xlabel('Potential (V)');
            ylabel('Current (amp)');
            grid on;
            hold off;
        end
    end
```